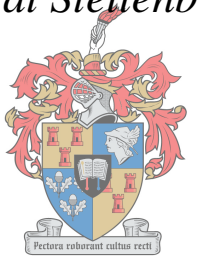
Detection and Tracking of Moving Objects Using Stereo Vision Cameras

Thesis presented in partial fulfilment of the requirements for the degree of Master of Engineering (Electronic) in the Faculty of Engineering at Stellenbosch University



UNIVERSITEIT
iYUNIVESITHI
STELLENBOSCH
UNIVERSITY

Author: *CE Roelofse*

Supervisor: *Dr CE van Daalen*

Department of Electrical and Electronic Engineering

Faculty of Engineering

Stellenbosch University

March, 2018

UNIVERSITEIT·STELLENBOSCH·UNIVERSITY
jou kennisvennoot • your knowledge partner

Plagiaatverklaring / *Plagiarism Declaration*

1. Plagiaat is die oorneem en gebruik van die idees, materiaal en ander intellektuele eiendom van ander persone asof dit jou eie werk is.
   Plagiarism is the use of ideas, material and other intellectual property of another's work and to present is as my own.

2. Ek erken dat die pleeg van plagiaat 'n strafbare oortreding is aangesien dit 'n vorm van diefstal is.
   I agree that plagiarism is a punishable offence because it constitutes theft.

3. Dienooreenkomstig is alle aanhalings en bydraes vanuit enige bron (ingesluit die internet) volledig verwys (erken). Ek erken dat die woordelikse aanhaal van teks sonder aanhalingstekens (selfs al word die bron volledig erken) plagiaat is.
   Accordingly all quotations and contributions from any source whatsoever (including the internet) have been cited fully. I understand that the reproduction of text without quotation marks (even when the source is cited) is plagiarism.

4. Ek verklaar dat die werk in hierdie skryfstuk vervat, behalwe waar anders aangedui, my eie oorspronklike werk is en dat ek dit nie vantevore in die geheel of gedeeltelik ingehandig het vir bepunting in hierdie module/werkstuk of 'n ander module/werkstuk nie.
   I declare that the work contained in this assignment, except where otherwise stated, is my original work and that I have not previously (in its entirety or in part) submitted it for grading in this module/assignment or another module/assignment.

March 2018

ABSTRACT

Detection and tracking of moving objects (DATMO) is one of the core components necessary for autonomous navigation of a robot. The robot requires measurements of its environment, and uses these measurements to construct a representation of the dynamic objects in its environment. Once dynamic objects have been detected, the robot can use their locations and movement for autonomous functionality such as navigation, or collision prediction and avoidance. Of particular interest are vision-based sensors such as cameras, due to the amount of information they give about the environment. However, the amount of information causes difficulty regarding the interpretation and workable utilisation of the information.

This thesis outlines a systematic approach for robust estimation of states for DATMO using stereo vision cameras. The mathematical basis of the camera geometry is derived. These include the camera projection transform, camera parameters, and epipolar geometry. Image-features are obtained from both the left and right cameras and are matched. These matched image-feature pairs are triangulated to form 3D measurements of the moving objects in the robot's environment. These 3D measurements are then used to filter the state estimates of the moving objects. Popular feature detection algorithms are expounded and investigated. ORB, KAZE, and A-KAZE are chosen for implementation and comparison. Factors pertaining to feature detection and matching, such as subpixel accuracy and matching strength, are weighed in light of a desired robust implementation.

The data association problem, that is which objects in the environment caused which measurements, is addressed. Methods used to address the problem, such as global nearest neighbour, probabilistic data association, and multiple hypothesis tracking (MHT), are examined. A multiple hypothesis tracking solution that uses Bayesian statistics is used in order to reliably associate measurements to objects in the robot's environment. Necessary approximations to the MHT approach are made and justified. The approximations result in a first-order approximation and a Gaussian mixture density description. The issue of unbounded associations is addressed and managed with techniques that remove, approximate, or prevent unnecessary state estimates.

Algorithms are tested using the KITTI dataset in Python. LiDAR is used to evaluate the results of the algorithm. The computational cost of the algorithm is the biggest issue highlighted by the results. This is due to the complexity of the multiple hypothesis tracking solution and the large number of image-features used to ensure robust and reliable functionality. The results of the thesis demonstrate that there is philosophical conflict between the requirement of robust estimation on the filtering side, and the large number of measurements required from camera images. The complexity increases with the number of measurements, but many measurements are needed in order to provide a reliable representation of the environment from camera images.

UITTREKSEL

Deteksie en volging van bewegende voorwerpe (DATMO) is een van die kernkomponente wat nodig is vir outonome navigasie van ’n robot. Die robot benodig metings van die omgewing en gebruik hierdie metings om ’n voorstelling van die dinamiese voorwerpe in sy omgewing te maak. Sodra dinamiese voorwerpe gevolg is, kan die robot hul posisies en beweging gebruik vir outonome funksies soos navigasie, of botsing-voorspelling en vermyding. Van belang is visie-gebaseerde sensors soos kameras, gegee die hoeveelheid inligting wat hulle beskikbaar maak oor omgewing. Die hoeveelheid inligting veroorsaak egter probleme met betrekking tot die interpretasie en werkbare benutting van die inligting.

Hierdie thesis beskryf ’n sistematiese benadering vir robuuste afskatting van toestande vir DATMO deur gebruik te maak van stereo-visie kameras. Die wiskundige basis van die kamera geometrie word afgelei. Dit sluit in die kamera projeksie transformasie, kamera parameters en epipolêre geometrie. Image-features word ontdek uit beide die linker en regter kameras en word geassosieer met mekaar. Hierdie geassosieerde image-feature-pare word gebruik om 3D-metings van die bewegende voorwerpe in die robot se omgewing te kry. Hierdie 3D-metings word dan gebruik om die toestandafskattings van die bewegende voorwerpe te filter. Die mees populêre feature algoritmes word uiteengesit en ondersoek. ORB, KAZE en A-KAZE word gekies vir implementering en vergelyking. Faktore wat verband hou met feature-opsporing en assosiasie, soos subpixel-akkuraatheid en assosiasie betroubaarheid, word geweeg in die lig van die verlangde robuuste implementering.

Die data-assosiasie probleem, dit wil sê watter voorwerpe in die omgewing veroorsaak watter metings, word aangespreek. Metodes wat gebruik word om die probleem aan te spreek word ondersoek. ’n MHT oplossing wat Bayesiese statistieke gebruik, word gebruik om metings betroubaar te assosieer met voorwerpe in die robot se omgewing. Noodsaaklike benaderings tot die MHT-benadering word gemaak en geregverdig, wat lei tot ’n eerste-orde benadering en ’n Gaussiese mengsel digtheidsbeskrywing. Die probleem van oneindige meting assosiasies word aangespreek met tegnieke wat onnodige toestandafskattings verwyder, benader of voorkom.

Algoritmes word getoets met behulp van die KITTI datastel in Python. LiDAR is gebruik om die resultate van die algoritme te evalueer. Die berekeningskoste van die algoritme is die grootste probleem wat deur die resultate uitgelig word. Dit is as gevolg van die kompleksiteit van die MHT oplossing en die groot aantal image-features wat gebruik word om robuuste en betroubare funksionaliteit te verseker. Die resultate van die thesis wys dat daar filosofiese konflik is tussen die vereiste van robuuste afskatting op die filterkant en die groot aantal metings wat van kamerabeelde vereis word. Die kompleksiteit neem toe met die aantal metings, maar baie metings is benodig om ’n betroubare voorstelling van die omgewing uit kamerabeelde te verseker.

TABLE OF CONTENTS

List of Figures

LIST OF TABLES

NOMENCLATURE

This section contains abbreviations and mathematical notations used in this thesis.

1. Abbreviations

A-KAZE Accelerated KAZE

AOS Additive Operator Splitting

BRIEF Binary Robust Independent Elementary Features

CPHD Cardinalised-PHD

DATMO Detection And Tracking of Moving Objects

div Divergence

DoG Difference of Gaussian

EKF Extended Kalman Filter

FAST Features from Accelerated Segment Test

FED Fast Explicit Diffusion

GM-PHD aussian Mixture Probability Hypothesis Density

GM Gaussian Mixture

GNN Global Nearest Neighbour

GPU Graphics Processing Unit

IMU Inertial Measurement Unit

IW Inverse Wishart

JPDA Joint Probabilistic Data Association

KF Kalman Filter

KL Kullback-Leibler

LDB Local Difference Binary

LoG Laplacian of Gaussian

M-LDB Modified-Local Difference Binary

MHT Multiple Hypothesis Tracking

MTT Multiple Target Tracking

oFAST Oriented FAST

ORB Oriented FAST and Rotated BRIEF

PDA Probabilistic Data Association

PHD Probability Hypothesis Density

rBRIEF Rotated BRIEF

RFS Random Finite Set

RTK Real Time Kinematic

SIFT Scale-Invariant Feature Transform

SLAM Simultaneous Localisation And Mapping

SURF Speeded-Up Robust Features

UKF Unscented Kalman Filter

2. Mathematical Notation

a	Scalar
$\mathbf{a}$	Vector
$\mathbf{A}$	Matrix
$\mathbf{x}_k$	Vector at timestep k
$\mathbf{x}_{1:k}$	Collection of vectors from timestep 1 to timestep k
X_k	Set at timestep k, unless defined otherwise
$X_{1:k}$	Set of collected sets from timestep 1 to timestep k
$\underline{\mathcal{G}}(\cdot)$	Vector function
$\boldsymbol{\mu}$	Gaussian mean
$\boldsymbol{\Sigma}$	Gaussian covariance
$\mathcal{N}(\cdot, \cdot)$	Define Gaussian distribution
$\overline{\mathbf{p}}'$	Homogeneous coordinate of pixel position
$\overline{\mathbf{p}}$	Homogeneous coordinate of state space position
$\mathbf{0}$	Vector of zeros
I	Identity matrix
$I_{KL}(\cdot\|\|\cdot)$	KL divergence
D^2_{MAL}	Squared Mahalanobis distance
$\chi^2(k)$	Chi squared distribution with k degrees of freedom
$\widetilde{\cdot}$	Merging approximation of quantities
$p(\mathbf{x})$	Probability density function of $\mathbf{x}$

1. INTRODUCTION

In the last 50 years, academia and business industry have been working toward realising a more autonomous robotic future. Robotics itself have been widely utilised and have even taken over certain industries. A prominent example of such an industry is the manufacturing industry, where robotics is extremely useful given the repetitive mundane nature of tasks. However, attempting to realise reliable autonomous robots has proven to be difficult and a more complex problem than anticipated. These robots need to function independent of human assistance in a dynamically changing environment with moving objects in the environment. Navigating independently in such an unstructured dynamic environment requires methodologies that are robust and reliable. The problem is further complicated by the ego (self) motion of the robot. All measurements are obtained at different positions in the environment and relating these measurements to one another requires the location of the robot to be known. Autonomous functionality requires robust techniques for sensor data filtering and reliable logic that governs autonomous decision making.

1.1 Autonomous Functionality

Sensor data is required in order to give a computer system an abstracted representation of its environment. This representation is abstracted by nature since it is not a direct exhaustive representation of the environment, and therefore reductionistic. This representation needs to be interpretable in such a way as to yield useful understanding of what is present in the environment, and therefore be functional, that is facilitate interaction with the environment. This interaction could be a range of desired applications such as navigating through the environment, or reliably predicting what might change in the environment. Autonomous functionality is a complex problem given the difficulty of robustly and reliably functioning in a dynamically changing environment.

Cameras have been of particular interest as sensors given the amount of data available in vision-based sensors. However, they have posed difficult challenges. One of the difficulties of processing vision-based data is that it is not self-evident how to process and interpret vision-based data in a way that results in a useful representation of the environment. Instead

of making direct state space data available, such as position and velocity, the appearance of things in the environment is instead available. The other issue is the computational burden of interpreting and processing the amount of data present. Regardless of the difficulties associated with cameras, they are still of interest for the purpose of autonomous functionality given the amount of information present. In fact, the ever-increasing capabilities and advancements of processors have gradually made vision-based sensors more relevant and viable, and continue to do so.

1.2 Probabilistic Robotics: Bayesian Statistics

The required usable representation needs to describe the states of moving objects of interest, such as their position and velocity. If these states are known, then they can be used to facilitate autonomous functionality. The robot could use the information to predicate where moving objects would be in the future, and navigate accordingly. However, the issues of sensor noise and robot localisation error result in uncertainty in the resultant states of moving objects. This uncertainty reduces the reliability of the environment's representation. These issues are addressed with probabilistic techniques and the approach known as probabilistic robots.

The challenge of autonomous functionality is approached with probabilistic methods. These methods are used because they model the inherent uncertainty both in the environment and in the measurements of the environment. Probability density functions are used to characterise measurement, process, and state estimate uncertainty. These probabilistic representations can be used in a Bayesian framework to produce estimates of the hidden states of moving and stationary objects [68], along with the robot's pose (position and orientation). These resultant estimates can be used to track moving objects and facilitate autonomous functionality. A Bayesian statistical approach is predicated on Bayes theorem [68] described by Equation 1.1, and can be derived using conditional distributions. This theorem is the basis of statistical inference. Given the prior distribution over random variable $\mathbf{x}$, and the conditional distribution of the measurement $\mathbf{z}$ given $\mathbf{x}$, the likelihood function $p(\mathbf{z}|\mathbf{x})$, the posterior distribution $p(\mathbf{x}|\mathbf{z})$ can be determined.

Given the likelihood function $p(\mathbf{z}|\mathbf{x})$, prior distribution $p(\mathbf{x})$, and the marginal distribution of the data $p(\mathbf{z})$, the posterior distribution $p(\mathbf{x}|\mathbf{z})$ can be determined. The posterior distribution is proportional to the product of the likelihood function and the prior distribution since $p(\mathbf{z})$ functions as a normalisation constant, that is

$$p(\mathbf{x}|\mathbf{z}) = \frac{p(\mathbf{z}|\mathbf{x})p(\mathbf{x})}{p(\mathbf{z})} \propto p(\mathbf{z}|\mathbf{x})p(\mathbf{x}). \tag{1.1}$$

Figure 1.1 demonstrates a simple two-dimensional $\mathbf{x} = [x_1,\ x_2]^T$ example of Bayesian inference, where the three ellipses represent one standard deviation contour plots of joint bivariant

Gaussian distributions over the $\mathbf{x} \in \mathbb{R}^2$ space. The prior belief (blue) about $\mathbf{x}$ is updated with a measurement (green) and results in a posterior belief (red). The Bayesian statistical framework facilitates the inherit uncertainty, in measurements and states, in order to yield resultants that represent the degree of the robot's certainty. Once moving objects' states and measurements are represented by random variables, they can be cast into a Bayesian framework. The resultant distributions are then used to facilitate autonomous functionality, but with the added benefit of having a measure of certainty about the environment that the robot is functioning in.

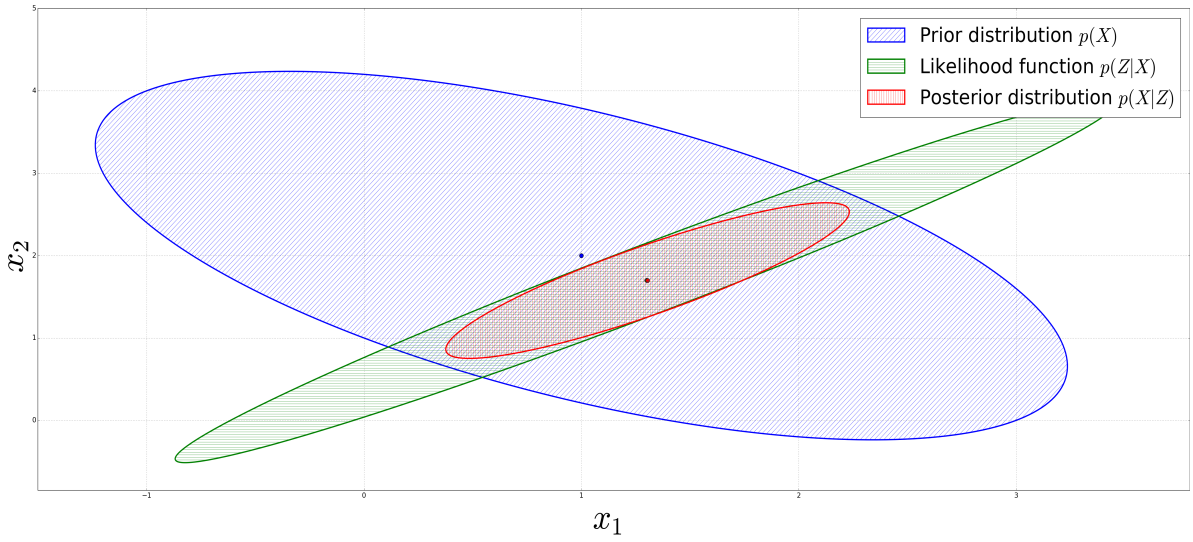


Figure 1.1: Two-dimensional joint Gaussian Bayesian inference example.

Applications of Bayesian Filtering: SLAM and DATMO

Two of the more popular design approaches that facilitate autonomous operation are SLAM (simultaneous localisation and mapping) and DATMO (detection and tracking of moving objects). These frameworks, given the discussed inherent uncertainty, are approached and modelled with Bayesian statistics.

The objectives of SLAM are twofold [65], [66]. First a map estimate of the robot's environment is created. This map is created by using measurements of stationary landmarks in the environment and the robot is localised with respect to the landmarks. Stationary landmarks are used to create the map due to their reliability as fixed points with little process noise or modelling error [75]. Moving objects have more relative uncertainty in their state estimates, and would cause increased uncertainty in the robot's localisation. The second objective is to localise the robot (the robot's pose) using the map estimate without the use of inertial sensors. Figure 1.2 shows a Bayesian network of the SLAM process. The network visually illustrates how different random variables cause or influence other random variables with arrows indicating the nature of causation. $\mathbf{x}_k$ indicates the robot pose, $\mathbf{z}_k$ the measurements, and $\mathbf{m}$ the

map, which consists of the estimated landmark locations. u_k denotes the robot commands that influence the next states of the robot. The prediction-update posterior result derived from Bayes' rule of conditional distributions is expressed by [76]

$$p(\mathbf{x}_k, \mathbf{m}|Z_k, U_{k-1}) \propto p(\mathbf{z}_k|\mathbf{x}_k, \mathbf{m}) \int p(\mathbf{x}_k|\mathbf{x}_{k-1}, u_{k-1}) p(\mathbf{x}_{k-1}, \mathbf{m}|Z_{k-1}, U_{k-2}) d\mathbf{x}_{k-1}. \quad (1.2)$$

The hidden map $\mathbf{m}$ and states $\mathbf{x}_k$ are estimated simultaneously with the assumption that the map is static, that is the landmarks constituting the map are stationary. Capitalised letters indicate the collection of all of its lower case counterparts from $k = 0$ up until the current subscript $Z_k = \{\mathbf{z}_0, \mathbf{z}_1, ..., \mathbf{z}_k\}$. The resultant posteriors can be used to facilitate autonomous functionality, given that the robot has an understand of where it is in the environment.

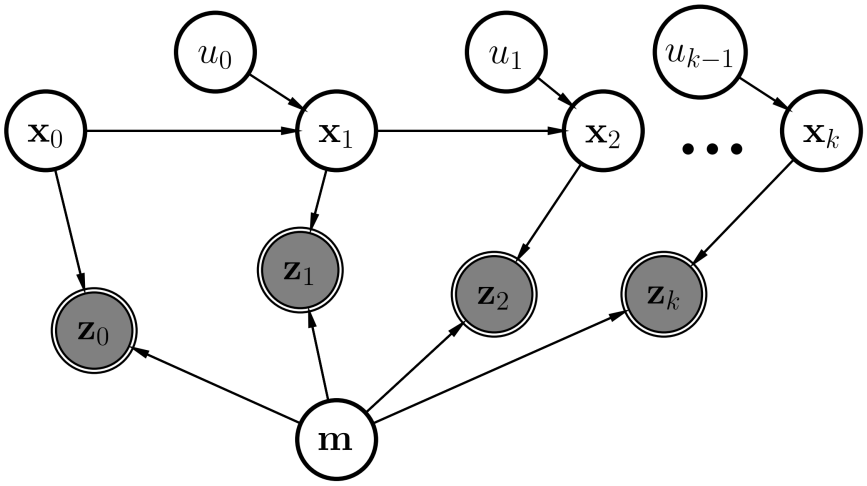


Figure 1.2: Bayesian network for SLAM. Shaded nodes indicating observed states and non-shaded nodes indicating hidden states. Subscripts indicating the discrete time instance.

The objective of DATMO is simply detecting and tracking moving objects and can be used in a variety of frameworks, including or excluding inertial sensors. DATMO can be used in conjunction with SLAM, and can be advantageous for the purposes of SLAM, given that DATMO can determine which objects are dynamic and therefore function as a form of outlier detection [74]. By determining which tracked objects are dynamic, the SLAM process can exclude dynamic tracked targets from candidacy as stationary SLAM landmarks. Petrovskaya has stated that DATMO can be classified into three different frameworks, traditional, model-based, and grid based DATMO [56]. Traditional DATMO focuses on estimating states in a Bayesian context with data segmentation and association techniques, that is determining which objects in the environment caused which measurements. Model-based DATMO performs inference directly on the sensor measurements without segmentation and association. Grid based DATMO constructs a grid representation of the dynamic environment of the robot.

Traditional DATMO is of interest due to its general Bayesian formulation and state inference in the state space.

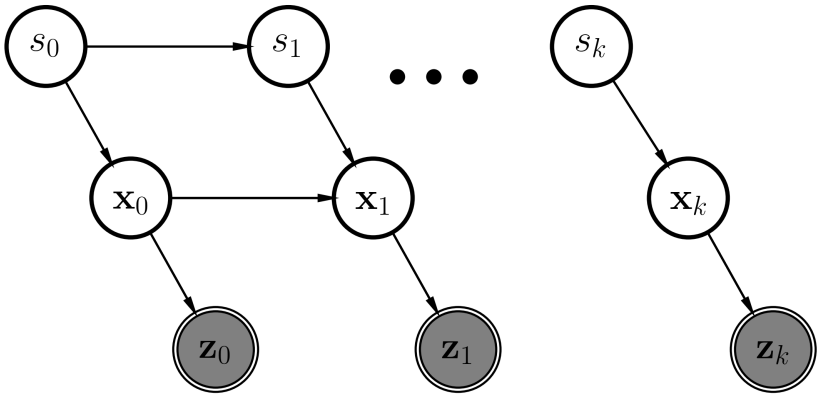


Figure 1.3: Bayesian network for DATMO. s_k indicates a discretely distributed random variable and denotes the motion model of a moving object, which governs the evolution of its states.

Figure 1.3 depicts a Bayesian network for traditional DATMO. The depiction deals with the issue of both state estimation and dynamic model estimation. $\mathbf{x}_k$ denotes the states of tracked moving objects, and $\mathbf{z}_k$ the measurements of the states. Any given moving object could be operating under different dynamics that govern its future states, for example a constant velocity model or a constant acceleration model. The depicted formulation estimates these dynamics and uses the dynamic model estimation to estimate the states. The estimation of model dynamics could increase the propagation accuracy of states and as a result increase the accuracy of the state estimates. Equation 1.3 describes the estimation of the state and motion mode of a moving object [76], that is

$$p(\mathbf{x}_k, s_k|Z_k) = p(\mathbf{x}_k|s_k, Z_k)\ p(s_k|Z_k). \tag{1.3}$$

Using Bayes' rule, the problem can be divided into a *state inference* $p(\mathbf{x}_k|s_k, Z_k)$ problem and a *model learning* $p(s_k|Z_k)$ problem [76]. The estimated model is used in the state inference process. In order to alleviate some of the computational burden, a *model selection* learning approach is typically used. Model selection chooses from a finite set of possible model dynamics modelled by the discrete random variable s_k.

This section illustrates how the formulation of a state estimation problem using Bayesian statistic can be used. This formulation models the inherit uncertainty in the process of state estimation. Resultant state estimates of tracked moving objects can be used to facilitate autonomous functionality such as path planning and collision prediction. SLAM was addressed in order to illustrate how DATMO can contribute to SLAM landmark candidacy filtering, and therefore be appropriated for more than the standard moving object tracking.

1.3 Cameras as Sensors: Approach and Limitations

Cameras are used for DATMO in a variety of configurations and use different image processing techniques. Cameras can be configured into mono, stereo, or multi-vision set-up. Stereo or multi-vision allows for depth information to be extracted, while mono-vision does not. Depth information for mono-vision can be obtained if the camera moves, assuming a relatively static environment, creating a synthetic baseline. Depth estimation accuracy (for any configuration) decreases as the baseline distance between cameras decrease, and decreases as the true depth of objects increase. Obtaining depth information of objects is required in order to estimate the states of an object or landmark in the environment.

In order for depth estimates to be obtained, corresponding points in at least two different camera images, with overlapping field of view of the scene, need to be found and correctly matched. Chosen points are referred to as *image features*. Typically an image feature consists of two things. First, a *keypoint* which is a location in the image plane with an orientation. Second, a *descriptor* which describes the appearance of the feature and is typically a function of the local gradients around the keypoint location. Salient pixel gradient and intensity characteristics are popular for interest point inspection, that is interest for feature usage. Interest points are inspected for feature candidacy due to their repeatability, that is the frequency with which they appear in multiple images over time. Their repeatability is attributed to their salient pixel gradient and intensity characteristics. Feature descriptors, describing points in the environment, are matched between two images at a given synchronised time, and as a result the depth of the point into the scene can be determined. The accuracy with which the depth can be determined is dependent on the accuracy with which a feature keypoint is localised. Given that an image has a finite resolution, subpixel localisation techniques are typically used to obtain more accurate feature keypoint locations.

During the feature matching phase, using the feature descriptors, there is always a chance that an incorrect correspondence can be made, that is a mismatched feature pair which leads to an incorrect representation of the environment. This issue can be mitigated to a certain extent by using a variety of techniques. Some of these techniques include using thresholds for the similarity of a potential match. Others include only using correspondences that are found to match both ways between image pairs, that is matching pairs that are found both when starting with the left image and search for matches in the right image, and vice versa. The geometrical constraints introduced by compensating for the pose of the individual cameras are also used to constrain the missed match issue.

Figure 1.4 is a simple representation of feature points in the world projected into the camera image plane. The dimensionality is reduced and only objects in the field of view of the camera are detected. These feature points are localised and described for the purpose of tracking and

matching between different cameras. These matches are triangulated in the environment and used in a Bayesian inference filtering framework in order to obtain estimates of the states of the detected objects.

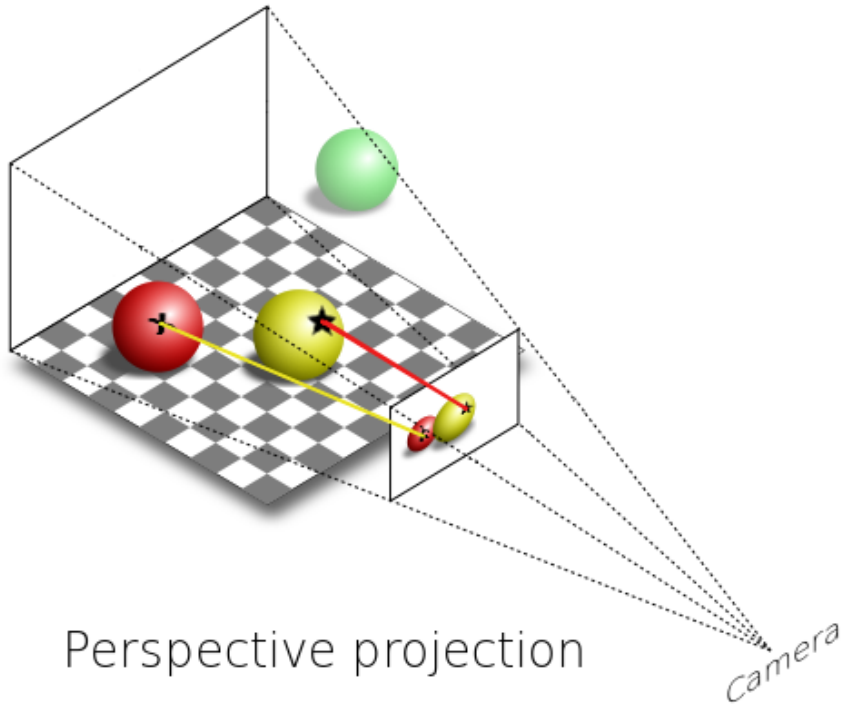


Figure 1.4: Camera projection from the 3D world, that is state space, to the image plane, that is measurement space. Image reproduced with permission from Rougier [2].

1.4 Thesis Scope

This thesis focuses on robustly and reliably obtaining and processing (filtering) measurements of moving objects using stereo vision cameras, in order to track and represent the activity present in the environment. The resultant representation could be used to facilitate a number of autonomous functionality. Various techniques used to generate measurements from camera sensor data is inspected with the focus on feature detection. The reliability of these techniques is of interest. Filtering these measurements in order to yield state estimates is done in a Bayesian filtering framework, which models the inherit uncertainty, as stated. The resultant state estimates give a measure of the reliability of these state estimate locations. The issue of determining which moving object generated which measurement, mentioned only briefly, is explored. This measurement causation problem is solved by determining which technique that addresses this issue is the most robust to error. Different feature detection techniques are used and compared in order to inspect their reliability as robust methods for DATMO.

2. LITERATURE REVIEW

This section contains a literature review about topics that relate to the application of tracking dynamics objects with cameras. These include camera modelling, stereo geometry, feature detection, measurement filtering, and data association strategies. The model used to describe the transformation of a 3D point to a 2D point in the image plane is derived. Once the camera model for an individual camera is derived, stereo vision geometry is expounded. This geometry is ultimately used for the purposes of determining the position of a 3D point in the environment, given a matched pair of features from the two stereo cameras. Determining the location of this 3D point is required in order to track a moving point in the environment.

Various feature detection algorithms are explored and investigated. Once points are found, they are matched using feature descriptors. The match can then be triangulated into the environment, resulting in the 3D state space location of the detected point. The issue of determining which measurements where caused by which moving objects needs to be addressed. The most popular techniques are evaluated and inspected.

2.1 Camera Model

Using cameras as sensors in tracking applications requires a mathematical foundation. This foundation should describe the mathematical transform that occurs when an object is within a camera's field of view. This transform is from the state space (the world) to the measurement space (the camera's image plane). In this section the mathematical model of a single camera is first derived, and then extended to stereo vision.

2.1.1 Pinhole Camera Model

This subsection focuses on the fundamentals of the popular *pinhole camera model* [58]. This foundation is canonically used in literature relating to any work that uses the camera projection transform. All camera parameters that are relevant to the required transform are modelled using the pinhole camera model as the foundation. After the pinhole camera model has been covered, it will be used to derive the transform of a 3D point in state space to its image

projected point in the camera, the measurement space. This transform is a function of the intrinsic camera parameters.

Intrinsic Camera Parameters

The pinhole camera models the aperture (small opening through which light passes) of the camera as a single point, and as a result the only light that enters the camera is light that passes through this pinhole point, called the *optical centre*. The pinhole modelling choice is an apt approach given that the aperture of a camera is necessarily small, since larger aperture sizes can result in more image blurring. This blurring is due to the fact that light from the same 3D point would intersect the image plane at multiple points. These intersection points increase as the aperture size increases. Using the pinhole model, light passes through the optical centre and creates a vertically inverted image on the *image plane* a distance of f behind the optical centre. This distance f is called the *focal length*. Analogously, an image plane a distance f in front of the camera (instead of behind) can conceptually be used. This image plane in front of the camera produces a non-inverted image. Both Figures 2.1 and 1.4 depict a visualisation of an image plane in front of the optical centre.

Figure 2.1 depicts the geometry used to derive the camera transformation model from the pinhole camera model [58]. The 3D Cartesian axis denotes the position of the camera with the origin being the optical centre. Y_C point downward below the camera. The *optical axis* corresponds to the Z_C axis and points in the direction that the camera is facing, that is the camera lens opening. This axis is referred to as the camera coordinate system.

A point $\mathbf{p} \in \mathbb{R}^3$ described in camera coordinates is projected onto the image plane and consequentially creates a point $\mathbf{p}' \in \mathbb{R}^2$. Notice how the projection line (which is the light ray) in Figure 2.1 is projected toward the optical centre (pinhole), and the intersection of the projection line and the image plane creates point $\mathbf{p}'$ a distance f from the optical centre. The camera projection transform results in a reduction of dimensions, that is

$$\mathbf{p}' = \underline{\mathcal{G}}(\mathbf{p}),\ \underline{\mathcal{G}} :\ \mathbb{R}^3 \to \mathbb{R}^2. \tag{2.1}$$

As a result, in the mono vision case, the full 3D state cannot be recovered if one starts with the projected point $\mathbf{p}'$, unless multiple cameras are used.

The transform $\underline{\mathcal{G}}$ can be derived using simple geometry of similar triangles that describes proportional triangle side lengths. Figure 2.2 is a two dimensional slice of Figure 2.1's model along the Y_C-axis, that is Figure 2.2 is in the $X_C Z_C$-plane. x_p is the projection's horizontal position (in pixels) in the image plane measured from the principal point $\mathbf{c}$ in Figure 2.1.

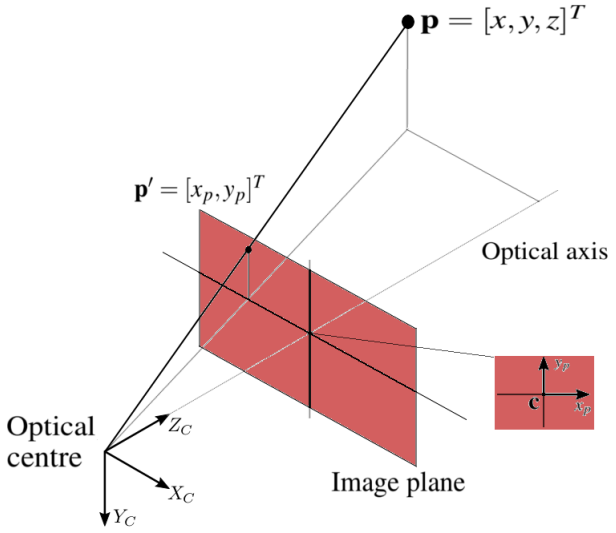


Figure 2.1: Pinhole camera projection model. $\mathbf{c}$ is the principal point where the optical axis intersects with the image plane. The figure is based on a diagram created by Botha [33].

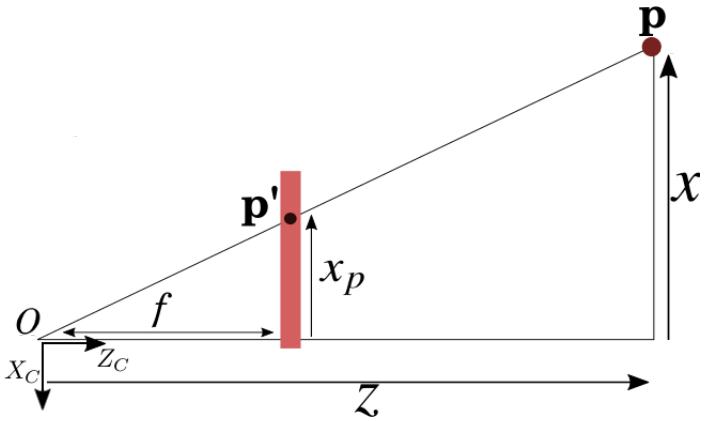


Figure 2.2: $X_C Z_C$-plane of camera projection model. $\mathbf{p}'$ is located in the image plane, denoted by the pink bar.

The desired transform can be expressed as

$$\frac{x_p}{f} = \frac{x}{z} \Rightarrow x_p = f\frac{x}{z}. \tag{2.2}$$

Equation 2.2 is derived using proportional triangle sides with f in units of pixels. Similarly,

$$\frac{y_p}{f} = \frac{y}{z} \Rightarrow y_p = f\frac{y}{z} \tag{2.3}$$

can be derived using a $Y_C Z_C$-plane section, y_p being the projection's vertical position (in pixels) in the image plane.

These transforms are traditionally expressed in matrix notation as

$$z\,\overline{\mathbf{p}}' = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \overline{\mathbf{p}}, \tag{2.4}$$

using homogeneous coordinates with $\overline{\mathbf{p}}' = [x_p,\ y_p,\ 1]^T$ and $\overline{\mathbf{p}} = [x,\ y,\ z,\ 1]^T$. This expression allows for the transform to be expressed in matrix form, which is not possible using Cartesian coordinates since the formulation does not allow for product and division operations. However, using homogeneous coordinates results in a matrix notation with $z \neq 0$, the depth of the 3D point, which scales the resultant projection point $\overline{\mathbf{p}}'$, as expressed in Equation 2.4.

At this point the transform of a 3D point in state space to measurement space has been derived. This derivation is facilitated by the pinhole camera model. Next some of the physical camera parameters that influence the transform will looked at and modelled. These parameters need to be modelled in order to result in an accurate projection transforms that is representative of the physical projection process.

The focal length f was the first of the mentioned parameters and is used to derive the projection transform. These parameters form part of the pinhole camera model. Other parameters include the principal point location, the non-square factors, and the skewed pixel factor. The principal point $\mathbf{c} = (c_x,\ c_y)$ is the point where the optical axis intersects the image plane. This point is not necessarily at the origin, the middle of the image plane, but can be offset. This offset needs to be modelled and accounted for. The non-square factors α and β are used to compensate for pixels which are vertically or horizontally elongated (as rectangles). The skewed factor τ is used to compensate for any vertically skew (slanting) aligned pixel columns. Radial lens distortion can also be modelled and compensated for as well, but is neglected in this section. Radial distortion does not form part of pinhole camera model, but is required to transform an image of an actual camera to something representing the pinhole camera model. Adding these intrinsic factors to Equation 2.4 results in

$$z\,\overline{\mathbf{p}}' = \begin{bmatrix} \alpha f & \tau & c_x & 0 \\ 0 & \beta f & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \overline{\mathbf{p}} = \left[\mathbf{K}_I \mid \mathbf{0}\right] \overline{\mathbf{p}}\ ; \tag{2.5}$$

$\mathbf{K}_I$ is referred to as the *intrinsic camera matrix* [29], [40]. These parameters are typically found using self-calibration methods such as those implemented by OpenCV. A variant of Zhang's [82] method of planar calibration is implemented in OpenCV. This method uses a specific planar pattern with known dimensions and with known feature point correspondence, that is features from the specific pattern. The plane and its features are observed by the camera from different perspectives which are used to determine the parameters expressed in Equation 2.5. A certain minimum number of perspectives are required, but usually more are used,

resulting in an overdetermined problem. This is done in order to result in estimates that are more robust to noise.

External Camera Parameters

The external parameters of the camera can be characterised by the pose of the camera's body axis relative to an inertial (or world) axis. An orthonormal 3x3 rotation matrix $\mathbf{R}_W$ and a translation vector $\mathbf{t}_W$, describing the pose of the inertial axis origin in camera coordinates, can be used to transform a point described in the inertial coordinate system $\overline{\mathbf{p}}_W$ to a camera coordinate system $\overline{\mathbf{p}}$ (Equation 2.6). Therefore, if a 3D point is described in any coordinate system and the pose of this coordinate system relative to the camera is known, then the 2D projection point can be determined. Similarly, rotation matrix $\mathbf{R}_C$ and a translation vector $\mathbf{t}_C$, describing the pose of the camera coordinate system origin relative to an inertial coordinate system, can be used to transform a point described in the camera coordinate system $\overline{\mathbf{p}}$ to inertial coordinate system $\overline{\mathbf{p}}_W$, that is

$$\overline{\mathbf{p}} = \left[\begin{array}{c|c} \mathbf{R}_W & \mathbf{t}_W \\ \hline \mathbf{0}^T & 1 \end{array}\right] \overline{\mathbf{p}}_W \;\&\; \overline{\mathbf{p}}_W = \left[\begin{array}{c|c} \mathbf{R}_C & \mathbf{t}_C \\ \hline \mathbf{0}^T & 1 \end{array}\right] \overline{\mathbf{p}}. \tag{2.6}$$

This transformation of coordinate systems is required since the camera projection transform is derived by assuming a camera coordinate system representation, with 3D points described in the camera coordinate system. However, when dealing with implementation, $\mathbf{R}_W$ and $\mathbf{t}_W$ are not available, but instead $\mathbf{R}_C$ and $\mathbf{t}_C$, the camera or robot's pose, are available by using an inertial sensor such as an IMU. Therefore, the *extrinsic camera matrix* $\mathbf{K}_E$ is derived and described in terms of $\mathbf{R}_C$ and $\mathbf{t}_C$, since they are typically known. Looking at the descriptions in Equation 2.6, it follows that

$$\left[\begin{array}{c|c} \mathbf{R}_W & \mathbf{t}_W \\ \hline \mathbf{0}^T & 1 \end{array}\right] = \left[\begin{array}{c|c} \mathbf{R}_C & \mathbf{t}_C \\ \hline \mathbf{0}^T & 1 \end{array}\right]^{-1}, \tag{2.7}$$

and decoupling the pose into translation and rotation matrices results in

$$= \left[\left[\begin{array}{c|c} I & \mathbf{t}_C \\ \hline \mathbf{0}^T & 1 \end{array}\right] \left[\begin{array}{c|c} \mathbf{R}_C & 0 \\ \hline \mathbf{0}^T & 1 \end{array}\right]\right]^{-1} = \left[\begin{array}{c|c} \mathbf{R}_C & 0 \\ \hline \mathbf{0}^T & 1 \end{array}\right]^{-1} \left[\begin{array}{c|c} I & \mathbf{t}_C \\ \hline \mathbf{0}^T & 1 \end{array}\right]^{-1}, \tag{2.8}$$

with I denoting the identity matrix, which simplifies [9] to

$$= \left[\begin{array}{c|c} \mathbf{R}_C^T & 0 \\ \hline \mathbf{0}^T & 1 \end{array}\right] \left[\begin{array}{c|c} I & -\mathbf{t}_C \\ \hline \mathbf{0}^T & 1 \end{array}\right] = \left[\begin{array}{c|c} \mathbf{R}_C^T & -\mathbf{R}_C^T\mathbf{t}_C \\ \hline \mathbf{0}^T & 1 \end{array}\right] = \mathbf{K}_E. \tag{2.9}$$

Factoring in the intrinsic camera matrix, as stated in this subsection results in

$$z\,\overline{\mathbf{p}}' = \left[\mathbf{K}_I \mid \mathbf{0}\right] \overline{\mathbf{p}}. \tag{2.10}$$

Expanding $\overline{\mathbf{p}}$ to be a function of a point described by any coordinate system, using Equation 2.6, results in

$$= \left[\mathbf{K}_I \mid \mathbf{0}\right] \left[\begin{array}{c|c} \mathbf{R}_W & \mathbf{t}_W \\ \hline \mathbf{0}^T & 1 \end{array}\right] \overline{\mathbf{p}}_W. \tag{2.11}$$

Substituting Equation 2.9 into Equation 2.11 results in

$$= \left[\mathbf{K}_I \mid \mathbf{0}\right] \left[\begin{array}{c|c} \mathbf{R}_C^T & -\mathbf{R}_C^T\mathbf{t}_C \\ \hline \mathbf{0}^T & 1 \end{array}\right] \overline{\mathbf{p}}_W \tag{2.12}$$

and

$$z\,\overline{\mathbf{p}}' = \left[\mathbf{K}_I \mid \mathbf{0}\right] \left[\mathbf{K}_E\right] \overline{\mathbf{p}}_W. \tag{2.13}$$

Equation 2.13 describes the camera projection transform in homogeneous coordinates, including intrinsic and extrinsic camera parameters [9]. Using this formulation requires an estimation of the intrinsic parameters and requires an inertial senor to obtain the extrinsic parameters. This formulation is for the mono vision case. Estimating the states of a moving object in 3D a requires depth information, which in turn requires multiple perspectives of the moving object. As a result, the projection transform needs to be expanded to the stereo vision case.

2.1.2 Stereo Geometry and Rectification

In practice, stereo image pairs are not necessarily aligned, that is projected points in one image that correspond to the projected points of the same 3D point in the second image are not horizontally aligned. This is due to their relative individual poses. A process known as stereo rectification is used to compensate for the relative pose of the cameras. Stereo rectifying

images with an overlapping field of view results in easier feature match searches. Once the images are stereo rectified, all corresponding projected point matches are horizontally aligned and need to be searched for along horizontal lines.

Stereo Rectification

Figure 2.3 shows a stereo camera setup with optical centres o_1 and o_2. Point $\mathbf{p}$ is projected to o_1 and o_2, resulting in $\mathbf{p}_1^{'}$ and $\mathbf{p}_2^{'}$. Optical centres are separated by baseline distance B. e_1 and e_2 are the points where B intersects the image planes and are known as *epipoles* [43]. The lines in the image planes (blue lines) from $\mathbf{p}_1^{'}$ to e_1 (first line) and from $\mathbf{p}_2^{'}$ to e_2 (second line) are known as *epipolar lines*. The first epipolar line represents possible positions for $\mathbf{p}_1^{'}$ in the image plane, given a fixed corresponding point $\mathbf{p}_2^{'}$. As $\mathbf{p}$ moves along the $\mathbf{p} - o_2$ line toward o_2, $\mathbf{p}_1^{'}$ moves along the epipolar line toward the epipole e_1. The correct correspondence matching point is located along the epipolar line, and as a result a search for the match is constrained along the epipolar line. This constraint is called the *epipolar constraint* [54].

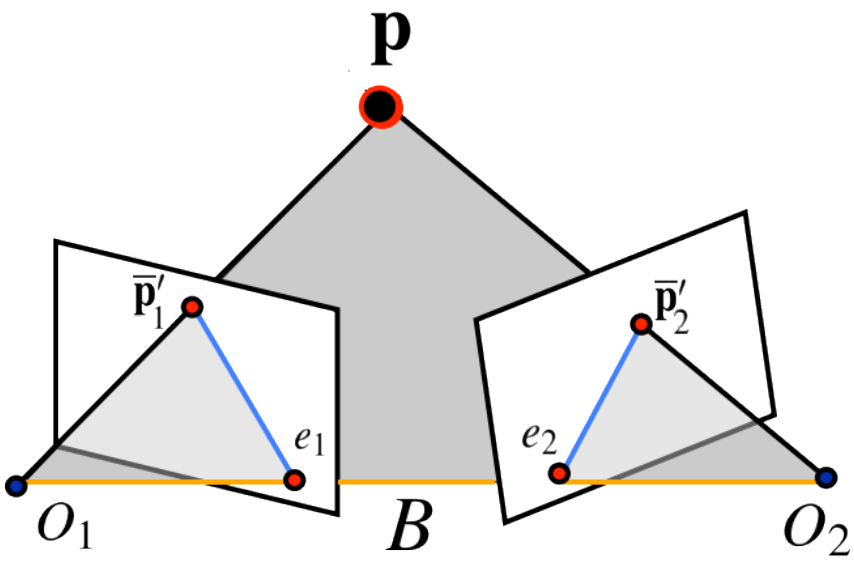


Figure 2.3: Stereo geometry displaying epipolar lines. A point is projected into two image planes, separated by baseline B, resulting in to projected points. These projection points are constrained along certain lines in the image planes. Image reproduced with permission from Savarese [1].

The process of rectification places the epipoles at infinity, and consequently the epipolar lines become parallel with the horizontal axis of the image planes. The image planes also become co-planar, situated in the same plane. The horizontal epipolar lines between the two images are aligned, placing each corresponding point at the same vertical position in the image planes, and along a horizontal line [1].

Figure 2.4 shows the result of rectification. Epipolar lines are horizontal and the epipoles are at infinity. The baseline intersects the shared plane in which the two image planes are now situated an infinite amount of time, that is the baseline lies within the plane. Corresponding points are located along the epipolar lines, which are now at the same vertical positions. This reduces the difficulty with which corresponding point matches need to be looked for. The resultant rectification is used in order to find image feature correspondences between rectified images. Practically, rectification is achieved by determining the pose of each camera relative to a planar surface with a known pattern and dimensions. Both cameras observe the planar surface with different poses. The known pattern produces strong feature responses with known correspondences between cameras. These correspondence matches between the two cameras of the same planar surface are used to determine the relative pose the cameras, and these poses are used to compensated for image alignment. Stereo rectification functions are also implemented in OpenCV.

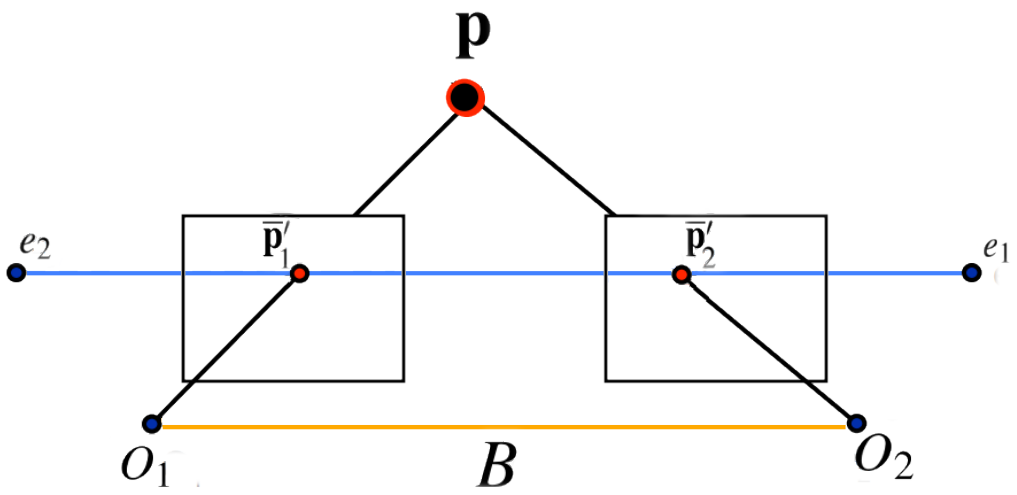


Figure 2.4: Stereo geometry displaying horizontal epipolar lines after rectification. Image reproduced with permission from Savarese [1].

The process of feature detection still needs to be addressed, but before feature detection processes are addressed, the stereo projection transform needs to be determined. This transform describes how a matched feature pair transforms from state space to measurement space, and vice versa, the reverse being referred to as *triangulation*, since the 3D point is triangulated into the state space using the matched pair. If the reader recalls, this reverse operation is not possible in the mono vision case since the mono vision case results in a reduction of dimensions.

Stereo Geometry

With images stereo rectified, the transform from 3D Cartesian camera coordinates, in meters, to image plane pixel positions, in pixels, can be derived. x_{pL} denotes the horizontal pixel position of $\mathbf{p} = [x,\ y,\ z]^T$'s projected position in the left image plane, and y_{pL} denotes the vertical pixel position of $\mathbf{p}$'s projected position in the left image plane. A four dimensional vector of a matched feature pair, $[x_{pL}, y_{pL}, x_{pR}, y_{pR}]$, is reduced to a three dimensional vector, $[x_{pL}, x_{pR}, y_p]$, given that $y_{pL} = y_{pR} = y_p$ after stereo rectification. The transform can be derived using similar methods as the mono vision case.

Figure 2.5 is a $X_C Z_C$ plane section of a stereo vision camera setup, using pinhole camera models. The principal point offset is neglected in the figure. o_1 and o_2 are the optical centres of the cameras. The origin of the Cartesian axis, describing $\mathbf{p}$, is located at o_1. The derivation is approached by using geometry of similar triangles, two equations from the $X_C Z_C$ plane section depicted in Figure 2.5, and one equation from a $Y_C Z_C$ plane section. The resulting equations are described by

$$x_{pL} = \frac{fx}{z} + c_x, \tag{2.14}$$

$$x_{pR} = \frac{f(x-B)}{z} + c_x, \tag{2.15}$$

and

$$y_p = \frac{fy}{z} + c_y. \tag{2.16}$$

The triangulation process describes the reverse of this transform, transforming a feature matched pair in the two image planes to a 3D position defined by

$$x = \frac{B(x_{pL} - c_x)}{x_{pL} - x_{pR}}, \tag{2.17}$$

$$y = \frac{B(y_p - c_y)}{x_{pL} - x_{pR}}, \tag{2.18}$$

and

$$z = \frac{fB}{x_{pL} - x_{pR}}. \tag{2.19}$$

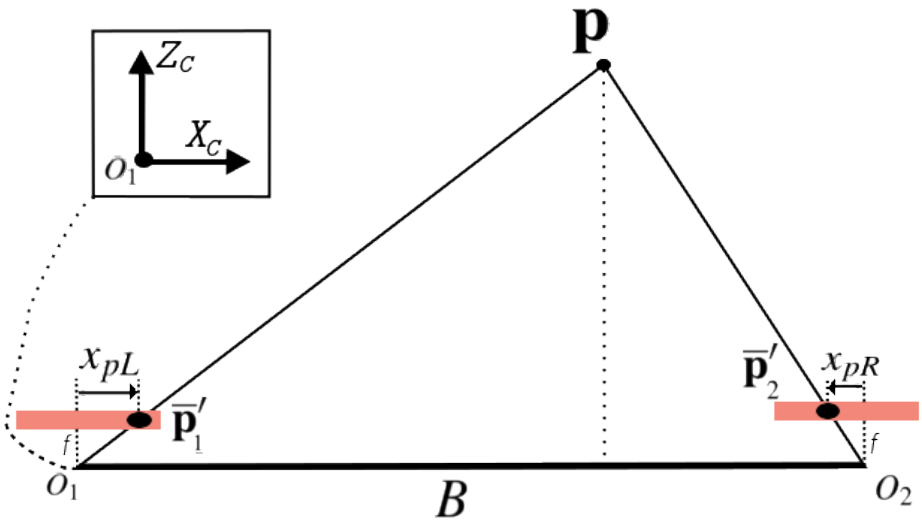


Figure 2.5: Stereo geometry, $X_C Z_C$-plane of camera projection model. The 3D point is projected into both the left and right stereo rectified cameras creating a corresponding match that can be triangulated back into the 3D space.

The triangulation equations can be used to determine the 3D position of a point in the environment. This point could be of a moving object, and as a result can be used to track a point in the 3D space. These image-feature points are determined by using feature detection techniques. Once features are detected, their descriptors are matches between stereo rectified images. Once matched, the feature matched pair can be triangulated into the environment.

2.2 Feature Detection

The derived camera model and transforms provide the framework in which pairs of matched points can be triangulated into the 3D space and vice versa. However, how to process and interpret the images themselves in a way that results in reliable and robust points is a separate problem. Image feature detection is one of the more popular methods of using visual sensor measurements. Feature detection searches for salient gradient or intensity behaviour and inspects them as interest points, that being interest for feature candidacy. Corner features are of interest due to their repeatability and distinctness, that is the relative reliability with which they will be appearing and be detectable over time.

These features consist of a *keypoint* location, with an orientation, and a *descriptor* describing the appearance of the feature. Descriptors are a function of the local gradients or intensities around keypoints and are used to describe a feature in a distinctive manner. Features are merely points in images, that when matched can be triangulated to a 3D point in the environment. As a result, they do not represent the full physical extent of moving objects. However, tracking a point located on a moving object is sufficient in order represent that there is in fact

a moving object present in the environment. Moving objects are assumed to be rigid, for example a vehicle, and therefore in most cases every point that could be tracked on the object would have the same velocity, given the rigid assumption. As a result, the assumption is that the velocity of a 3D point mass located on a moving object is representative of the movement and location of the object given the rigid assumption, but it is not representative of the object's extent. Therefore, features can be used to track objects, even though they lack extent representation.

Desirable attributes of features are rotation invariance and scale invariance. If a descriptor is rotation and scale invariant, then the descriptor can be matched even if the scene or camera rotates (rotation), or if the camera moves closer or further away from the moving object (scale). Some of the most popular algorithms used to detect features from images include SIFT, SURF, KAZE, and FAST, which are, amongst others, expounded in this section.

2.2.1 Early Feature Detectors

This subsection investigates two of the early feature detectors that are used to find corner points in an image. These are not considered as candidates for the thesis, but contain critical techniques that are used by the more modern feature detectors.

Harris Corner Detector

The Harris corner detector is the one of the oldest means of finding feature points in an image. Corners in an image have large pixel intensity gradients in both the horizontal x and vertical y directions. In order to detect these gradients, the differences of pixel intensities in a certain window w are inspected, as expressed by

$$E(x,y) = \sum_u \sum_v w(u,v)[\mathbf{I}(u+x, v+y) - \mathbf{I}(u,v)]^2. \tag{2.20}$$

The size of the corner in the image would determine how large the window would need to be. $E(x,y)$ is the sum of the weighted square of pixel intensity differences in a given window. $\mathbf{I}(x,y)$ denotes the pixel intensity of a grey scale image at *(x,y)*, and *w(u,v)* is a window function (box or Gaussian) that is swept over the image. The content in this subsection is reproduced from OpenCV documentation [6] and Derpanis's paper on the Harris corner detector [27].

Equation 2.20 is simplified by approximating $\mathbf{I}(u+x, v+y)$ with a first-order Taylor series expansion, as described by

$$\mathbf{I}(u+x, v+y) \approx \mathbf{I}(u,v) + \mathbf{I}_x(u,v)x + \mathbf{I}_y(u,v)y, \tag{2.21}$$

with the underlying assumption being that the windowing shifts are small. $\mathbf{I}_x$ and $\mathbf{I}_y$ denotes partial derivatives of image $\mathbf{I}$ with respect to the variable indicated by their subscript. This assumption results in

$$E(x,y) \approx \sum_u \sum_v w(u,v)[\mathbf{I}(u,v) + \mathbf{I}_x(u,v)x + \mathbf{I}_y(u,v)y - \mathbf{I}(u,v)]^2, \tag{2.22}$$

and can be further condensed to

$$E(x,y) \approx \sum_u \sum_v w(u,v)[\mathbf{I}_x(u,v)x + \mathbf{I}_y(u,v)y]^2. \tag{2.23}$$

Equation 2.23 can be written in matrix form as

$$E(x,y) \approx \begin{bmatrix} x & y \end{bmatrix} \mathbf{M} \begin{bmatrix} x \\ y \end{bmatrix}, \tag{2.24}$$

with $\mathbf{M}$ defined as

$$\mathbf{M} = \sum_u \sum_v w(u,v) \begin{bmatrix} \mathbf{I}_x^2 & \mathbf{I}_x\mathbf{I}_y \\ \mathbf{I}_x\mathbf{I}_y & \mathbf{I}_y^2 \end{bmatrix} = \begin{bmatrix} \langle \mathbf{I}_x^2 \rangle & \langle \mathbf{I}_x\mathbf{I}_y \rangle \\ \langle \mathbf{I}_x\mathbf{I}_y \rangle & \langle \mathbf{I}_y^2 \rangle \end{bmatrix}. \tag{2.25}$$

Angled brackets $\langle \cdot \rangle$ denote averaging, which is caused by the summation and is a function of the type of window (weightings) used. The resulting matrix is referred to as the *Harris matrix* and its elements are the x and y derivatives of a patch (segment) of an image, isolated my means of a window function. In order to maximise the function $E(x,y)$, large variations in both x and y are required, which can be evaluated by inspecting the Harris matrix's eigenvalues, $\lambda_{1,2}$. Large eigenvalues indicate large deviations in the vector $[x\ ,\ y]^T$. Small eigenvalues indicate a planar (flat) region, that is a low gradient monochrome region. A single large eigenvalue indicates a single large gradient, an edge, and two large eigenvalues indicate a corner. The ratio of the two eigenvalues needs to be smaller than a certain threshold in order for the area to be regarded as a reliable corner.

Eigenvalue decomposition is computationally expensive and as a result

$$R = \lambda_1\lambda_2 - k(\lambda_1 + \lambda_2)^2 = |\mathbf{M}| - k\,\mathrm{Trace}^2(\mathbf{M}) > \lambda_{min} \tag{2.26}$$

has been suggested by Derpanis [27], which is a function of the eigenvalues, but is computationally more efficient. R gives a measure of the ratio of the eigenvalues. The constant k has been calculated empirically for typical applications (in the range of 0.05 – 0.15). As a result, the determinant and trace operations are used, instead of eigenvalue decomposition, and if the result is larger than a determined threshold the interest point is regarded as a reliable corner.

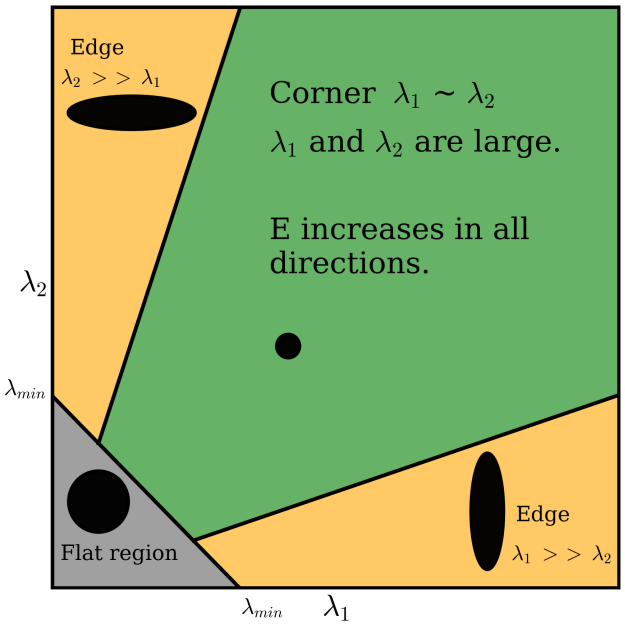


Figure 2.6: Eigenvalue classification illustration for Harris corner detector. This illustrates the classification of an inspected area as either being a plane (grey, small $|R|$), being an edge (orange, $R < 0$), or being a corner (green, large R). The ellipses are visual representations of the eigenvalue ratios. Large eigenvalues with small ratios are desirable.

The result of this classification process is a function of the Harris matrix's eigenvalues. The classification process can be visually represented by Figure 2.6. The Harris corner detector as a stand-alone method is not scale invariant, and is not itself used as a feature detector since it does not build a feature descriptor. However, the methods used in this early interest point detector plays an important role in further developments.

Shi-Tomasi: Good Features to Track

Shi and Tomasi, in good features to track [64], augmented Equation 2.26 resulting in

$$R = \min(\lambda_1, \lambda_2) > \lambda_{min}, \tag{2.27}$$

and demonstrated that their adjustment yields a better threshold scoring function than Equation 2.26. Equation 2.27 simply states that both λ_1 and λ_2 need to be larger than a certain threshold λ_{min}. Instead of determining a result that concerns itself with the ratio of the eigenvalues, both eigenvalues simply need to be larger than a specified threshold. This alteration can be visually represented by a similar eigenvalue plot depicted by Figure 2.7. Evidently, both Harris and Shi-Tomasi's methods result in similar mapping regions as portrayed by Figures 2.6 and 2.7, with Shi-Tomasi's method being a more reductionist approach, since it ignores some of the ratio complexity present in Derpanis's derivation [27].

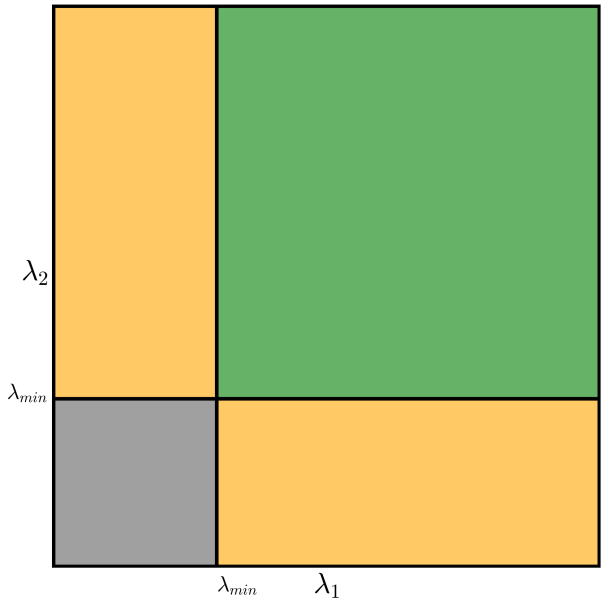


Figure 2.7: Eigenvalue classification illustration for Shi-Tomasi corner detector which is simpler than the Harris corner detector.

2.2.2 Modern feature detectors and formulations

While the early interest point detectors offer critical insight into feature detection strategies, they lack descriptors, and rotation and scale invariance. The modern feature detection techniques address these issues resulting in robust distinct (identifiable) feature points. In this section SIFT, SURF, KAZE, A-KAZE, FAST, and ORB feature detection techniques are explored and walked through in detail.

SIFT

SIFT (scale-invariant feature transform), introduced in 2004, is an algorithm that detects features represented by descriptors that are vectors of floats. This method is both scale and rotation invariant. It achieves these robust properties by creating a scale space of an image, often referred to as a pyramid, and searches for local extrema across pixel position and scale space. The content in this subsection is reproduced from Lowe's SIFT paper [44] and OpenCV documents [7].

The scale space is created by convolving the image with a Laplacian of Gaussian (LoG) kernel with different variances σ^2. Convolving an image with a Gaussian dependent kernel is often referred to a *blob detector*, and the result is a blurred image of lower effective resolution, that is not necessarily an image of less pixel dimensions, but of less spectral information which could be represented by an image of less pixel dimensions. The variance of the Gaussian determines the size of the blob, or object, that can be identified in the image, since it determines the

degree of blurring. The LoG is the Laplacian operator ∇^2 applied to a Gaussian $G(x,y,\sigma) = \frac{1}{2\pi\sigma^2}\exp(\frac{x^2+y^2}{2\sigma^2})$ as described by

$$\nabla^2 G(x,y,\sigma) = G_{xx} + G_{yy}, \tag{2.28}$$

where subscripts denote partial derivatives. Figure 2.8 displays a two dimensional LoG. $\sigma^2\nabla^2 G$ is used to create a scale-normalised LoG, with σ^2 denoting the scale factor. This normalisation factor is required to create scale invariance.

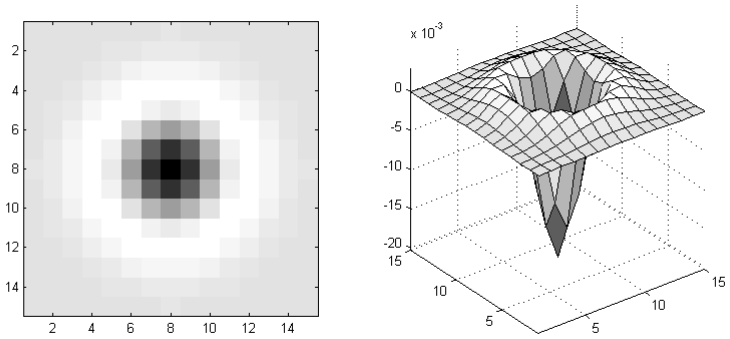


Figure 2.8: Two dimensional LoG [11] kernel used to detect blobs in a image. On the left is a 2D top view indicating pixel intensity, while on the right is a 3D view of the function.

An approximated kernel, called the difference of Gaussian (DoG), is often used in practice instead of using the LoG, since the DoG is more computationally efficient and is a sufficient mathematical approximation of the LoG kernel. The DoG approach involves filtering an image with Gaussians of different variances and subtracting them from each other. This operation is effectively a bandpass filter, preserving certain spacial frequencies, determined by the variances used, and results in a blob detector. This DoG $D(x,y,\sigma)$ is described as

$$D(x,y,\sigma) = (G(x,y,k\sigma) - G(x,y,\sigma)) * \mathbf{I}(x,y) \tag{2.29}$$

and therefore

$$= G(x,y,k\sigma) * \mathbf{I}(x,y) - G(x,y\sigma) * \mathbf{I}(x,y), \tag{2.30}$$

since convolution is distributive. Therefore, the subtraction can occur before or after the convolution process. k represents a scaling factor that scales one of the Gaussian standard deviations with respect to the other in Equations 2.29 and 2.30, resulting in a DoG.

Lowe [44] shows that the relationship between the DoG and scale-normalised LoG can be understood from the appropriated heat diffusion equation described by

$$\sigma\nabla^2 G = \frac{\partial G}{\partial \sigma} \approx \frac{G(x,y,k\sigma) - G(x,y,\sigma)}{k\sigma - \sigma}, \tag{2.31}$$

with the partial derivative approximated on the right side. The DoG can then be expressed by

$$G(x,y,k\sigma) - G(x,y,\sigma) \approx (k-1)\sigma^2\nabla^2 G. \tag{2.32}$$

The $(k-1)$ factor is constant over all scales and as a result does not influence the locations of the extrema.

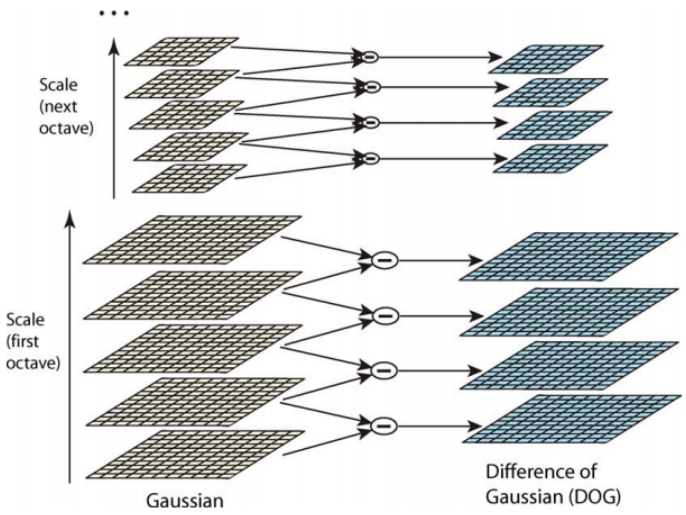


Figure 2.9: Visual representation of the scale space, the dimension introduced by blurring the original image, and the construction of the DoG. The image on the left are progressively more blurred from the bottom up. The differences between these images are used to form the DoG. Image reproduced with permission from Lowe [44].

Figure 2.9 illustrates the created scale space and the resultant DoGs. Local extrema need to be searched for in the resulting images by comparing neighbouring pixels across both scale (below and above in scale) and pixel position. This results in a list of potential interest points at a specific scale that might be robust enough for practical use. If interest points are deemed to be robust enough, they will be regarded as features.

The second part consists of locating the interest points (points of interest for feature candidacy) with subpixel accuracy and filtering out interest points that are located on edges. If the subpixel position intensity is less than a certain threshold, then it is also rejected since they would be too sensitive to noise to be robust.

In order to locate an interest point with subpixel accuracy, a quadratic function is fitted to DoG scale space function. This is achieved by obtaining a second order (quadratic) Taylor series expansion of the scale space function D, as described by

$$D(\mathbf{x}) \approx D(\mathbf{x}_{loc}) + \left[\frac{\partial D^T}{\partial \mathbf{x}} \bigg|_{\mathbf{x}=\mathbf{x}_{loc}} \right] \mathbf{x} + \frac{1}{2} \mathbf{x}^T \left[\frac{\partial^2 D}{\partial \mathbf{x}^2} \bigg|_{\mathbf{x}=\mathbf{x}_{loc}} \right] \mathbf{x}, \tag{2.33}$$

with $\mathbf{x}_{loc} = [x_{loc}, y_{loc}, \sigma_{loc}]^T$ denoting the location and scale of the interest point. $D(\mathbf{x}_{loc})$ is evaluated at $\mathbf{x}_{loc}$ and denotes the constant Taylor series offset from the linearision point $\mathbf{x}_{loc} = (x, y, \sigma)$. The three-dimensional quadratic's local minima or maxima is located where the gradient of the function is zero, which results in a subpixel point.

The interest points need to be inspected and rejected if the they are located along edges. This filtering process is required because the DoG operation will also yield points along edges as potential feature points. The *Hessian matrix* $\mathbf{H}_{Hes}$, described by

$$\mathbf{H}_{Hes}(x, y, \sigma) = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}, \tag{2.34}$$

is used to perform this inspection at the interest point's location and scale in the scale space function D, with subscripts denoting partial derivatives. Similar to the Harris matrix, the eigenvalues of the matrix are inspected. If both eigenvalues are large, then it indicates large derivatives is all directions, a corner, while only one large (relatively large) eigenvalue indicates an edge. Just like the Harris matrix, the ratio of the eigenvalues needs to be smaller than a certain threshold to be regarded as a corner and not an edge. The determinant and trace operations can also be used instead of eigenvalue decomposition, like the Harris case described by Equation 2.26, resulting in a more computationally efficient implementation.

Now that points (interest points deemed to be robust) have been identified, with edge responses and noise sensitive points filtered out, these new keypoints need to be described. First an orientation is calculated for each keypoint. This orientation is calculated at each keypoint's respective scale so that calculations are scale invariant, and will be used to achieve rotation invariance. In order to calculate the orientation, the gradient magnitudes *m(x,y)* and orientations $\theta(x, y)$ are computed beforehand for each individual image in the scale space $\mathbf{L}(x, y)$ (arbitrary image in the scale space) using the differences of pixel intensities as described by

$$m(x, y) = \sqrt{\Big(\mathbf{L}(x+1, y) - \mathbf{L}(x-1, y)\Big)^2 + \Big(\mathbf{L}(x, y+1) - \mathbf{L}(x, y-1)\Big)^2} \tag{2.35}$$

and

$$\theta(x,y) = \tan^{-1}\left(\frac{\big(\mathbf{L}(x,y+1) - \mathbf{L}(x,y-1)\big)}{\big(\mathbf{L}(x+1,y) - \mathbf{L}(x-1,y)\big)}\right). \tag{2.36}$$

The calculated gradient orientations in an area around the keypoint are divided into histogram bins. These orientations, depicted on the left side of Figure 2.10, are weighted according to their magnitude, and truncated using a Gaussian window. The largest histogram value/peak indicates the largest local gradient direction. The three closest values to the peak are used as sample points to fit a parabola to the local gradient function. This parabola is then used to interpolate the peak orientation value with increased accuracy, and as a result the keypoint is assigned with an orientation. Before the descriptor is calculated, all of the local gradients are rotated with respect to the determined orientations, resulting in a rotation invariant descriptor.

So far, each keypoint has been assigned a location, scale, and orientation and is done in such a way that the keypoints are invariant to these parameters and to noise. The next step is to assign a descriptor for each keypoint in order to identify the keypoint across multiple images, between stereo images or between images in a time sequence. This descriptor is a function of the local gradients at the keypoint's position and scale. The left side of Figure 2.10 shows the gradient magnitudes and orientations calculated with the blue circle representing the Gaussian weighting window. The Gaussian window adds more relative weight to the gradients that are closer to the localised position of the keypoint.

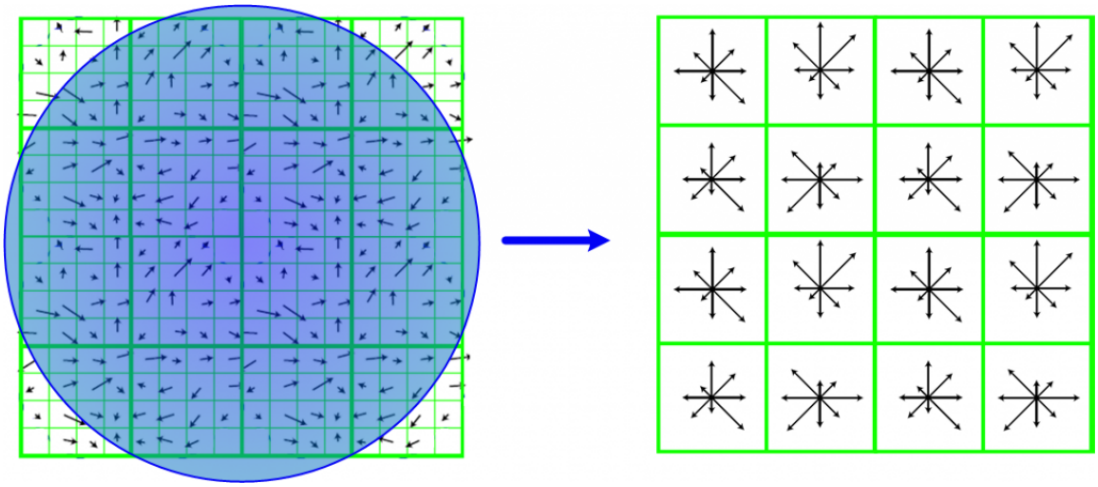


Figure 2.10: Visual illustration of a SIFT feature descriptor [4]. The gradient arrows are all added together and weighed in their respective subregion of the image into discrete 45^o locations. Image reproduced with permission from Lowe [44].

The divided local gradient orientations are used to determine the descriptor as seen depicted by Figure 2.10. The length of each arrow corresponds to the sum of the gradient magnitudes in their respective directions within each image subregion (section). The directions are discretely

divided into 8 positions, every 45 degrees, as seen in the figure. The vectors are rotated by the orientation assigned to the keypoint, thereby making the descriptor rotation invariant. As seen in Figure 2.10, 4x4 subregions, comprising, in total, of a 16x16 sample array, are used. The right side of Figure 2.10 visually represents the descriptor of the keypoint. These 4 by 4 subregions with 8 discrete directions results in a 4x4x8 = 128-dimensional vector.

The SIFT feature detector has become a staple in computer vision due to its rotation and scale invariant subpixel features. However, the algorithm is computationally intensive. SURF features were created in order to alleviate some of the computational issues of SIFT.

SURF

SURF (speeded-up robust features) is a computationally more efficient feature detection algorithm created by H. Bay et al. in 2006 based on the SIFT algorithm. SURF is roughly three times faster than SIFT with slight degradation in accuracy, but still accurate enough to be comparable with SIFT [38]. Considering that the DoG approach, used in the SIFT framework, can be used to approximate the LoG kernel, Bay et al. [21] decided to take the approximation further by approximating the LoG with a box filter, as shown in Figure 2.11. The content is this subsection is reproduced from Bay et al. [21], Guerrero et al. [38], and OpenCV documentation [8].

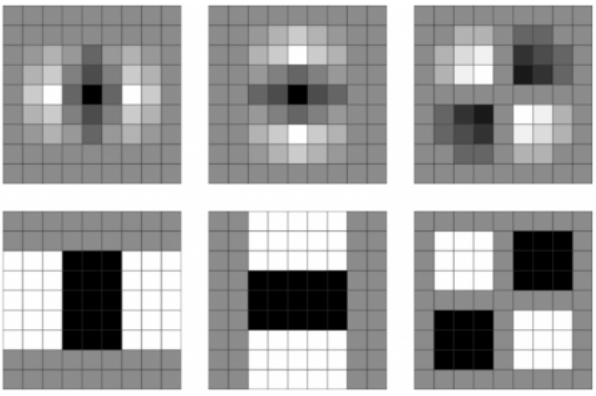


Figure 2.11: Box kernel approximation (in two dimensions) of the LoGs used in the SIFT framework. The approximation forms the part of basis of the SURF approach [12]. Image reproduced with permission from Tuytelaars [21].

The DoG, which is an approximation of the LoG, is used to create the scale space of image in the SIFT framework. SURF uses a box kernel approximation to create the scale space, which is a further approximation. Initially this approximation would seem computationally more efficient since a LoG does not need to be discretised for every scale, but the real computational advantage is that the convolution process is reduced to simple additions by using an integral image. An integral image is defined by

$$\mathbf{I}_I(x,y) = \sum_{i=0}^{i\leq x} \sum_{j=0}^{j\leq y} \mathbf{I}(i,j). \tag{2.37}$$

Every point *(x,y)* in the integral image is equal to the sum of all the pixel intensities in the upper left hand quadrant of the input image. Using a box kernel results in the same operations that are facilitated by the integral image. Figure 2.12 shows how the sum of the pixel intensities in a region Σ of the input image $\mathbf{I}(x,y)$ can be computed using the appropriate integral image $\mathbf{I}_I(x,y)$ values (A, B, C and D) with simple addition and subtraction, approximating the convolution process. Since the integral image does not change with scale, multiple box filtering convolutions can be performed in parallel for different scales, increasing the speed of the algorithm further.

The SURF algorithm uses the Hessian matrix to find interest points in the scale space function after the scale space has been made created using the box filter approximation. The Hessian matrix is described by

$$\mathbf{H}_{Hes}(x,y,\sigma) = \begin{bmatrix} \mathbf{L}_{xx} & \mathbf{L}_{xy} \\ \mathbf{L}_{xy} & \mathbf{L}_{yy} \end{bmatrix} \approx \begin{bmatrix} B_{xx} & B_{xy} \\ B_{xy} & B_{yy} \end{bmatrix} \tag{2.38}$$

with footnotes indicating partial derivatives with $\mathbf{L}(x,y,\sigma) = G(x,y,\sigma) * \mathbf{I}(x,y)$, G is a Gaussian kernel. Therefore $\mathbf{L}_{xx}$ is the LoG convolved with the input image. The LoG kernel is approximated with a box filter kernel $B(x,y,\sigma)$.

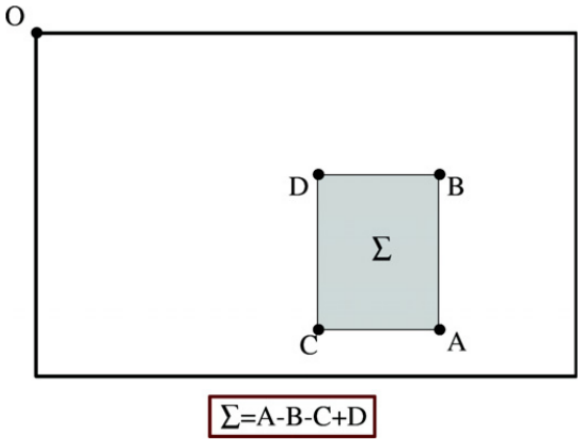


Figure 2.12: Illustration of how an integral image is used for box kernel convolution. Image reproduced with permission from Tuytelaars [21].

In order to identify a potential interest point, the determinant of the Hessian is inspected, since eigenvalue decomposition is a computationally expensive operation. The determinant of the Hessian matrix is equal to the product of the eigenvalues, as described by

$$|\mathbf{H}_{Hes}(x,y,\sigma)| = \lambda_1\lambda_2, \tag{2.39}$$

and the eigenvalues ($\lambda_{1,2}$) give an indication of the gradient deviation of the pixel intensities which determines if the point is a potential feature. The determinant is calculated using

$$|\mathbf{H}_{Hes}(x,y,\sigma)| = \mathbf{L}_{xx}\mathbf{L}_{yy} - \mathbf{L}_{xy}^2 \approx B_{xx}B_{yy} - (0.9B_{xy})^2. \tag{2.40}$$

The 0.9 factor was empirically determined by Bay et al. [21] to make the box filter kernel approximation more accurate.

SURF uses first order Haar wavelet (Figure 2.13) responses in the x and y directions in order to calculate a keypoint's orientation and descriptor. The depicted Haar wavelet responses can be calculated using an integral image, which allows for further computational savings since the integral image has already been calculated. The responses give an indication of the local gradients and are weighted with a Gaussian window. The responses are all added in individual subregions of a sliding orientation window of 60^o around the keypoint, creating a local dominant gradient vector in each region as the window slides around. Each of these local dominant vectors are compared and the largest is assigned as the keypoint's orientation as shown in Figure 2.14.

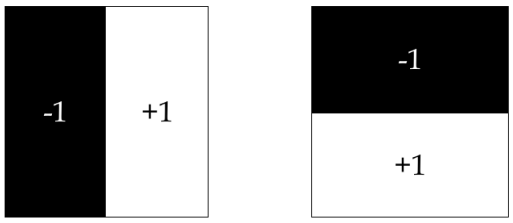


Figure 2.13: First order Haar wavelets. Image reproduced with permission from Tuytelaars [21].

The feature's descriptor is constructed by creating a vector $\mathbf{v}$ for each subsection around the keypoint. 4x4 subsections are used in total, and the wavelet responses in each subsection are added together to form a four-dimensional vector. These subsections are orientated according to the dominate keypoint orientation in order to make the resultant descriptor orientation invariant. This four dimensional vector $\mathbf{v}$ is made for each of the 4x4 subsections, and is described by

$$\mathbf{v} = \left(\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|\right). \tag{2.41}$$

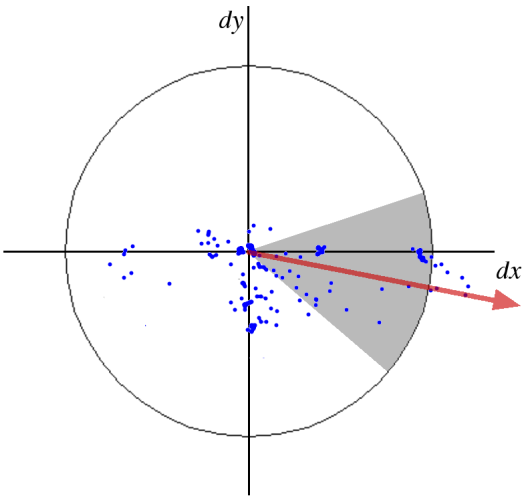


Figure 2.14: Determining the dominant orientation vector. The red arrow indicates the dominant orientation vector for the region under inspection, indicated by grey. The Blue dots represent the individual wavelet responses. Image reproduced with permission from Tuytelaars [21].

$d_{x/y}$ denotes the wavelet response with respect to the relative dimension, these responses are summed across the 2D pixel position at the scale the keypoint was located. These 4-dimensional vectors for each 4x4 section results in a total 4x4x4 = 64-dimensional keypoint descriptor.

SURF also uses the sign of the Laplacian in the feature matching stage. The sign of the Laplacian (positive or negative) indicates if a bright blob is located on a dark background or if a dark blob is located on a bright background. Matching is only considered between features if the two keypoints have the same sign of the Laplacian. The sign of the Laplacian is the trace of the Hessian matrix and therefore has already been calculated and can be used at no extra cost.

SURF offers faster feature detection, but with reduced performance. However, the results are still of such a nature that they are comparable with SIFT. Both SIFT and SURF constructs the scale space using linear diffusion techniques, that is Gaussian based blurring or approximations thereof. A new wave of feature detectors, such as KAZE and A-KAZE, are of interest due to their nonlinear diffusion techniques which results in a unique approach to creating the scale space.

KAZE

In 2012, Alcantarilla et al. [13] proposed a new algorithm for detecting and describing features called KAZE features [13]. KAZE also, like other algorithms, searches for features across both scale and pixel position, but constructs the image scale space in a different way.

Where algorithms like SIFT and SURF construct the scale space with linear diffusion filtering techniques, such as Gaussian blurring, KAZE constructs the scale space by using nonlinear diffusion filtering techniques. Content reproduced from Alcantarilla et al. KAZE paper [13].

Alcantarilla et al. [13] states that Gaussian blurring approaches blurs (filters) both noise and object details, such as object boundaries, to the same extent. As a result, the localisation accuracy of a feature keypoint is reduced, especially for features found at scales of larger degrees of blurring. Larger degrees of Gaussian blurring causes more smoothing of the boundaries of objects, and as a result the localisation accuracy is reduced, whereas nonlinear diffusion approaches preserves the boundaries of objects. Figure 2.15 displays the differences of linear and nonlinear diffusion filtering at various degrees of smoothing intensities. Notice particularly how the nonlinear approach gradually filters out details, like the shapes of branches within the boundaries of the trees, but preserves the outer boundaries of the trees in comparison with the linear approach.

Figure 2.15: Example illustrating the differences between linear and nonlinear diffusion filtering. The top row shows the result of Gaussian blurring at various degrees of intensity, while the bottom row shows the the equivalent nonlinear diffusion filtering. Image reproduced with permission from Alcantarilla [13].

The proposed framework uses a description of the evolution of an image **I**'s intensities through different scales (along the scale dimension or space) as the divergence of a flow function. This flow function, expressed as a partial derivative, is described by

$$\frac{\partial \mathbf{I}}{\partial t} = \text{div}\big[c(x,\ y,\ t) \cdot \nabla \mathbf{I}\big], \tag{2.42}$$

where div indicates the divergence operation. Alcantarilla et al. that t for the flow function in Equation 2.42 typically indicates time, but that in its appropriation in the desired framework it is used for scale, that is the derivative of **I** as it changes with scale, not time. The *conductivity* function that influences the flow function in a nonlinear fashion is expressed by

$$c(x,\ y,\ t) = g\big(\ |\ \nabla \mathbf{I}_\sigma(x,\ y,\ t\ )\ |\ \big) = g, \tag{2.43}$$

where $\mathbf{I}_\sigma$ is a slightly smoothed version of the original image that forms the basis of the scale space. Perona and Malik [55] proposed that the conductivity function should be a function of gradient magnitude in order to promote and retain edges through the filtering process. The KAZE paper proposes three different conductivity possibilities, two by Perona and Malik [55], and one by Weickert [77]. Perona and Malik proposes one that promotes edges and one that promotes larger regions as opposed to smaller ones. Weickert's proposition is expressed by

$$g = \begin{cases} 1 & ,\ |\nabla \mathbf{I}_\sigma|^2 = 0 \\ 1 - \exp(-\frac{3.315}{(|\nabla \mathbf{I}_\sigma|/k)^8}) & ,\ |\nabla \mathbf{I}_\sigma|^2 > 0 \end{cases}, \tag{2.44}$$

which ultimately is favoured for image scale space construction, with the k parameter adjusting the intensity of the filtering. This conductivity function promotes strong filtering on the sides of an edge, and has a weak response over an edge. This results in intraregional blurring, and preserves the boundaries of objects.

The partial differential equation, Equation 2.42, does not allow a closed form solution, and as a result, numerical methods are used. This has initially been an implementation limitation due to the computational burden of discretisation that is necessary. The discretisation step size needs to be small enough to ensure convergence over the scale space (dimension), but decreasing the step size for computational gain runs the risk of not converging. The proposed solution, by Weickert et al. [79], is additive operator splitting (AOS) which, along with the *Thomas algorithm* for solving tridiagonal systems of linear equations, yields stable scale space results for any step size. The AOS discretisation of Equation 2.42 is expressed by

$$\frac{\mathbf{I}^{k+1} - \mathbf{I}^k}{t_{k+1} - t_k} = \sum_{l=1}^{m} \mathbf{A}_l(\mathbf{I}^k)\ \mathbf{I}^{k+1}, \tag{2.45}$$

which can be used to iteratively solve for $\mathbf{I}^{k+1}$, as expressed by

$$\mathbf{I}^{k+1} = \left[I - (t_{k+1} - t_k) \sum_{l=1}^{m} \mathbf{A}_l(\mathbf{I}^k)\right]^{-1} \mathbf{I}^k, \tag{2.46}$$

where $\mathbf{A}_l$ is a matrix that encodes the image conductivities along each dimension.

KAZE detects its features, key points and descriptors, using similar methods used by SIFT and SURF, such as finding interest points using the Hessian matrix response over scale and space. KAZE uses float vectors as descriptors (as opposed to binary sequences). The most important contribution of the KAZE feature approach is the creation of an image scale space that results in more accurate localisation, especially at higher scales. This is due to the nonlinear approach, and its numerical implementation, that preserves the boundaries of objects at larger scales instead of blurring object boundaries such as Gaussian blurring techniques.

A-KAZE

A-KAZE (accelerated KAZE) is a binary descriptor variant and computational improvement of the KAZE framework [14]. A-KAZE features are also found, like in the KAZE framework, by inspecting a nonlinear scale space, created with the same flow function as Equation 2.42. A-KAZE uses *fast explicit diffusion* (FED, [78], [37]) to solve the partial derivative diffusion equation. FED outperforms AOS (used for KAZE), both in computational speed and effective flow function discretisation accuracy [80]. The content in this subsection is reproduced from Alcantarilla's A-KAZE paper [14]. Discrete sampling step sizes of the flow function is performed with varying incremental sizes, governed by

$$\tau_j = \frac{\tau_{max}}{2\cos(\pi\frac{2j+1}{4n+2})}, \tag{2.47}$$

which is performed for a predetermined number of cycles. τ_{max} sets the upper bound for the step sizes.

A-KAZE finds interest points by inspecting the constructed nonlinear scale space in the same way that KAZE does, which is similar to the way that SIFT or SURF does. The Hessian response, and eigenvalues, are inspected over the scale space and normalised with respect to scale. The ratio of eigenvalues is used as an indication of whether a corner, edge, or plane is under inspection.

A-KAZE describes its keypoints with binary sequences, as opposed to KAZE's float vectors. The technique is called modified local difference binary (M-LDB) and is a variant of the standard LDB descriptor [81] used by BRIEF [67]. This variant performs binary tests of the average pixel intensities, instead of the intensities of individual pixels. The averages of the horizontal and vertical derivatives of the area under inspection are also tested, which, in total, results in three binary tests per inspection. M-LDB is made rotation invariant by rotating the area under inspection according to the rotation of the discovered keypoint, and scale invariant by sub-sampling the scale space in increments that are a function of the keypoint's scale, instead of using all pixels below the scale position of interest.

While the methods addressed so far (SIFT, SURF, KAZE, and A-KAZE) use gradients to detect features, there are methods that use pixel intensities to detect features. Two of these methods, FAST and ORB, are covered before concluding this section on feature detection algorithms.

FAST

FAST (features from accelerated segment test) is a keypoint detector that was published in 2006. While SIFT and SURF are gradient based, FAST is pixel intensity based. As a re-

sult, it has been experimentally shown that FAST outperforms SIFT and SURF for images of planar surfaces (walls etc.) due to the lack of gradients. FAST has been shown to be significantly faster (as its name suggests) than SIFT and therefore more favourable for real-time applications, but lacks subpixel interpolation accuracy and a descriptor. The content in this subsection is reproduced from Rosten et al. [62], Viswanathan's summery [70], and OpenCV documentation [5].

Pixels are investigated for feature candidacy by testing their intensity relative to surrounding pixels. Pixel p, with an intensity of i_p, is inspected by comparing it with 16 neighbouring pixels on a Bresenham circle of radius 3 as seen in Figure 2.16. Pixel p is classified as a distinct keypoint if there exists a set of n ($n \leq 16$) contiguous pixels on the Bresenham circle that are either all brighter or darker than pixel p with a certain threshold t, $i_p > t + i_x$ or $i_p < i_x - t$. i_x denotes the pixel intensity of one of the neighbouring pixels that forms part of the set of 16 pixels under inspection. n was originally chosen to be 12 by Rosten et al. [62].

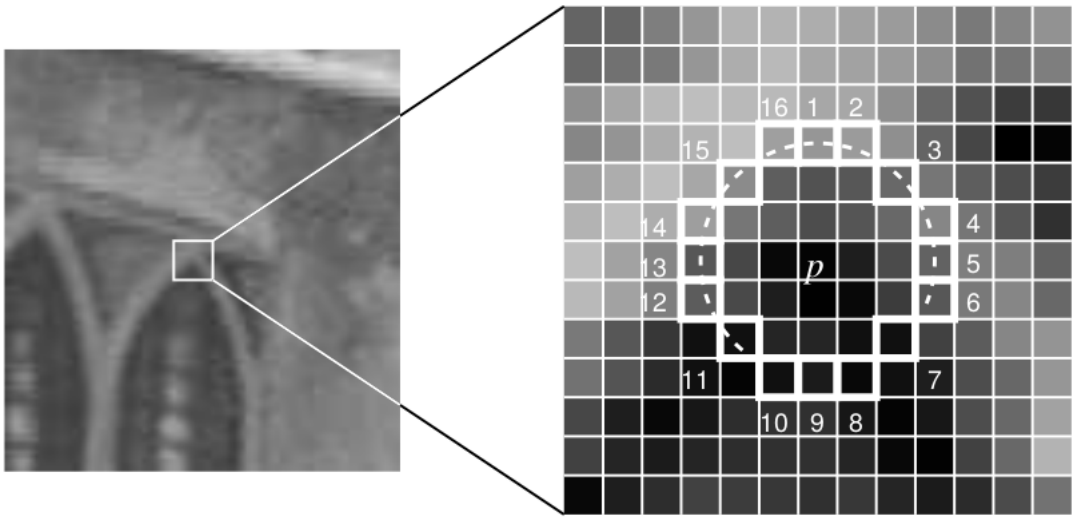


Figure 2.16: Illustration of the FAST approach. A pixel p under inspection with its 16 neighbouring pixels are displayed. Image reproduced with permission from Rosten [62].

An additional test is also used to pre-filter a large amount of non-corner pixels. The test consists of comparing only pixels 1, 5, 9 and 13 (Figure 2.16), and if at least three of those pixels are not brighter or darker with the mentioned threshold, then the point p is disregarded as an interest point. The full segment test is then performed on all the points that where not filtered out by the pre-filtering stage.

The FAST algorithm, as described so far, has several limitations that Rosten et al. [62] addressed with a machine learning approach and non-maximal suppression. The first problem is that the pre-filtering test runs the risk of prematurely removing desirable interest points. If

$n < 12$ is used as the minimum number of passed tests, then certain interest points will be filtered out by the pre-filtering stage that would have been regarded as a keypoint if they had been tested by the full segment test. The full segment test will not be performed on them since they have already been filtered out.

The second problem is that the ordering of the 16 neighbouring pixels is not optimal. The sequence in which the 16 pixels are inspected will influence the speed of the FAST algorithm. If only 4 pixels are required ($n = 4$) to pass the classification test as a corner, then the algorithm will be faster if all 4 required pixels just so happen to be inspected first, in comparison with a scenario where they just so happen to be inspected last of the 16 pixels. The third problem is that the FAST algorithm yields multiple interest points cluttered adjacent to each other, which is unnecessary since they are all likely responding to the same underlining corner in the image.

A machine learning approach is used to address the first and second problems. First a set of images are selected for training from the desired application domain. These images are examples of the type of images that will be inspected for corners. The FAST algorithm is executed on the whole set of images using an appropriate choice for n and threshold t. Next, every pixel x on the Bresenham circle, $x \in \{1,...,16\}$, is categorised into a class (or state) expressed by

$$S_{p\to x} = \begin{cases} d, & i_x \leq i_p - t \\ s, & i_p - t < i_x < i_p + t \\ b, & i_p + t \leq i_x \end{cases}, \tag{2.48}$$

where $S_{p\to x}$ indicates the state of x, i_x is the pixel intensity of x and i_p is the intensity of the pixel p under inspection. d, s, and b indicate the states darker, similar, or brighter respectively, and indicates the intensity state of x relative to p.

Determining $S_{p\to x}$ for x of all $p \in P$, P being the set of all pixels in the test images, divides P into 3 subsets P_d, P_s, and P_b. A boolean variable k_P is defined that is true if p is a corner and false if not. A decision tree classifier (ID3) is used to inspect the created subsets of P. The entropy of k_P is used as a measure of information gain (IG) to find the pixel x that yields the most information about the state of p (corner or non-corner). This inspection process is recursively applied to the subsets until the entropy is zero. This process determines the minimum amount of inspections of x that is needed in order to determine whether p is a corner or not.

The third problem, the cluttered adjacent responses, is addressed with non-maximal suppression where some of the adjacently located interest points are discarded. A score function v_{score} is calculated for interest points, and those with the lowest v_{score} are discarded. v_{score} is defined

as the sum of the absolute difference between all 16 pixels in a contiguous arc of the centre pixel p, and is described by

$$v_{score} = \max \begin{cases} \sum(\text{arc pixel value} - p), & \text{if } (\text{arc pixel value} - p) > t \\ \sum(p - \text{arc pixel value}), & \text{if } (p - \text{arc pixel value}) > t \end{cases}. \tag{2.49}$$

FAST tests integer located positions and as a result is not accurate to subpixel accuracy. Subpixel accuracy could be achieved by extra techniques, such as interpolating after performing a parabola curve fit at a detected keypoint. FAST is not rotation or scale invariant, which causes issues as the scene changes, or if the robot or objects move in the environment. FAST is used in the ORB framework and given orientation, resulting in rotation invariance, and a descriptor, resulting in scale invariance.

ORB

ORB (oriented FAST and rotated BRIEF) [63] is a feature detector that appropriates and augments FAST as a keypoint detector [62], and BRIEF (binary robust independent elementary features) [67], a binary keypoint descriptor, to create a feature detector. Rublee et al. [63] shows that ORB is about two orders of magnitude faster than SIFT [63]. These already well-established algorithms (FAST and BRIEF) where chosen for their relatively low computational cost. Rublee et al. augmented the BRIEF descriptor algorithm to be rotation invariant, since standard BRIEF descriptors are not rotation invariant, and they assign an orientation to the FAST keypoints, resulting in oFAST (oriented FAST). The content in this subsection is reproduced from Rublee et al.'s ORB paper [63].

The FAST algorithm is first augmented with a filtering process. The FAST algorithm has the potential to yield keypoints along edges, which is undesirable. Edge points are filtered with a Harris matrix response. All detected keypoints are ranked according to a Harris response, with corner responses being on the top of the list and edges at the bottom. A subset of determinant size, desired or required, is selected from the list and the rest of the listed points, consisting of mostly edges (depending on the chosen size), are filtered out. The second change to FAST's implementation is that a scale space of the original image is constructed, and FAST keypoints are detected and inspected across the constructed scale space.

Orientation is assigned to FAST features by using the *intensity centroid* response [61]. This approach assumes an offset between the corner centre $\mathbf{o}$ and the corner's centroid $\mathbf{c}$. A vector $\underline{\mathbf{oc}}$ from the centroid point to the centre is used as a measurement of orientation. The definition of the moment, used by the centroid, depending on the order p and q, is expressed by

$$m_{pq} = \sum_{x,\, y} x^p y^q \, \mathbf{I}(x,\, y), \tag{2.50}$$

which sums over the sub-image (patch) bounds of image $\mathbf{I}$ where the keypoint is located. The centroid location in the image patch around the keypoint is calculate by

$$\mathbf{c} = \left[\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}} \right] \tag{2.51}$$

and the orientation by

$$\theta = \text{atan2}(m_{01},\, m_{10}). \tag{2.52}$$

x and y are the horizontal and vertical pixel positions respectively.

BRIEF constructs a binary (bit) sequence descriptor by performing binary tests as a function of a image subsection (patch) p's intensities. p is the patch around the location of the discovered keypoint. p is smoothed before the binary tests are performed. The binary test is expressed by

$$\tau(p;\, \mathbf{x},\, \mathbf{y}) = \begin{cases} 1, & p(\mathbf{x}) < p(\mathbf{y}) \\ 0, & p(\mathbf{x}) \geq p(\mathbf{y}) \end{cases}, \tag{2.53}$$

and the descriptor vector is constructed using

$$f_n(p) = \sum_{1 \leq i \leq n} 2^{i-1}\, \tau(p;\, \mathbf{x}_i,\, \mathbf{y}_i), \tag{2.54}$$

which consists of n binary tests, where n is typically 256. $\mathbf{x}$ and $\mathbf{y}$ are both pixel position vectors in the patch surrounding the discovered keypoint.

BRIEF is augmented to be rotation invariant. Rublee et al. [63] demonstrate that BRIEF fails as a distinct identifiable descriptor for even small angular rotations of the region where the keypoint is located. The first change was coined as *steered BRIEF*. Steered BRIEF is the result of rotating the image patch using the orientation determined by the centroid response. The set of all n binary testing pairs are adding to a 2 x n matrix, as

$$\mathbf{S} = \begin{bmatrix} \mathbf{x}_1,\, \mathbf{x}_2,\, \ldots,\, \mathbf{x}_n \\ \mathbf{y}_1,\, \mathbf{y}_2,\, \ldots,\, \mathbf{y}_n \end{bmatrix}, \tag{2.55}$$

and rotated in accordance with θ, expressed by

$$\mathbf{S}_\theta = \mathbf{R}_\theta \mathbf{S}. \tag{2.56}$$

$\mathbf{R}_\theta$ is the 2x2 rotation matrix as a function of θ which is used to rotate the patch. The binary test operations are used on the resulting set of rotated points, as seen expressed by

$$g_n(p,\, \theta) = f_n(p) | (\mathbf{x}_i,\, \mathbf{y}_i) \in \mathbf{S}_\theta, \tag{2.57}$$

which now performs the operation $f_n(p)$ with $(\mathbf{x}_i,\ \mathbf{y}_i)$ being an element of the steered set $\mathbf{S}_\theta$.

An unfortunate consequence of the steered BRIEF approach is the introduced correlation between binary tests. Where BRIEF features are typically charactered by large binary sample test variances, steered BRIEF's binary tests are not, and hence are not as distinct. Rublee et al. [63] addressed the issue with a machine learning approach that chooses a subset of tests that are less correlated, that is tests that contribute more unique (new) information to a feature's representation (descriptor). The resulting method is called *rotated BRIEF* (rBRIEF), and the combination of oFAST and rBRIEF is called ORB.

2.3 Overview of Modern Feature Detectors

Section 2.2 outlines some of the most popular feature detectors. The details that underlying each of the feature detectors are expounded to show how the different techniques find and describe features. The individual methods and related details that constitutes each detector are shown to demonstrate how they add to the robustness and cost of each detector, or lack thereof. Of particular interest are the methods which are used to create the scale space, especially the new non-linear diffusion techniques which were first published to be used for feature detection as early as 2012.

Table 2.1 summarises some of the metrics of interest related to feature detection. The computational costs of feature extracts are difficult to compare since they vary between applications. It is easy enough to know, given their derivations, that SURF is computationally cheaper than SIFT, and A-KAZE cheaper than KAZE, but it is difficult to compare, for example, ORB and A-KAZE. Chien et al.'s paper [25] demonstrated that ORB is computationally cheaper than A-KAZE, while Pieropan et al.'s paper [57] demonstrated that A-KAZE is cheaper than ORB for 3 out of 4 tests that involved different image resolutions.

At this point, the stereo camera model has been derived which enables triangulation of matched points, between cameras, into the environment. Even though these points lack extent representation, they give an indication of the states of moving objects in the environment. Points need to be extracted from images in order to match and triangulate them. Feature detection algorithms are used to make use of these camera images and detect and describe points. These features can be detected and described with different methods, some more desirable than others. However, once features are detected, matched, and triangulated, they need to be filtered in such a way as to result in estimates of the states of moving objects. These estimates need to take into account the relative reliability of measurements and the reliability of already obtained state estimates. States and measurements are modelled with probabilistic representations (random variables), which gives a measure of the certainty of state estimates.

Table 2.1: Comparison of feature detection techniques inspecting various characteristics. Y = Yes, N = No, L = Linear. The computational costs are listed as relative comparisons, i.e. SIFT > SURF: SIFT is more costly than SURF.

	SIFT	SURF	KAZE	A-KAZE	ORB	FAST
Computational cost	>SURF	<SIFT	>A-KAZE	<KAZE	>FAST	<ORB
Subpixel accuracy	Y	Y	Y	Y	N	N
Descriptor	Float	Float	Float	Binary	Binary	None
Scale diffusion	L	L	Non-L	Non-L	L	None
Rotation invariance	Y	Y	Y	Y	Y	N
Scale invariance	Y	Y	Y	Y	Y	N

2.4 Single Target Tracking and Filtering

Once reliably repeatable features have been obtained and matched, they can be triangulated into the state space. These feature matches are themselves the measurements that can be used to determined where objects are and where they are moving in the environment, which is represented by the states of the objects. The states of interest are the position and velocity of an object, and the feature measurements give an indication of the position states. These measurements are, however, not perfect and have a degree of noise attached to them (measurement noise). The future states of a moving object could also be unpredictable due to disturbances (process noise). Issues also arise from incorrect feature matches. Trying to optimally estimate the states of a moving object, that is given the obtained and available information, is usually cast into a probabilistic framework. A probabilistic approach facilitates the modelling of uncertainty which is present in the estimation problem due to noise, and as a result, is not only an apt approach, but an effective one.

The *filtering problem* addresses the issue of determining the belief distributions of a time varying system's hidden (not directly observed) states by taking partial measurements of the system's states, that is measurements that are not necessarily of each individual state. The goal is to yield an optimal result which involves estimating a 3D point's states given the measurement and process noise. However, this approach does not address the issue of measurement correspondence, that is which measurement was caused by which object. The origins of measurements need to be known in order to make the correct associations. The following derivation assumes a single tracked object (or target) with known correspondence, but the issue of correspondence has to be addressed to track multiple moving objects in the environment.

Figure 2.17 displays a Bayesian network of a Markov process that represents the filtering problem, where $\mathbf{x}_k \in \chi_{space}$ represents the hidden states of the system (single object) at time

instant k on the *state space* χ_{space}, the space of all possible states (position and velocity). $\mathbf{z}_k \in \mathcal{Z}_{space}$ represents the partial noisy measurements of the hidden states of the system at time instant k on the *measurement space* $\mathcal{Z}_{space}$, the space of all possible measurements. u_{k-1} is the control input that influences the system's hidden states $\mathbf{x}_k$ at k.

Bayes Filter

The filtering problem is approached by determining the belief distribution over the states given the information presented. The analytical solution of the filtering problem, after the derivation, is known as the *Bayes filter*. The Bayes filter is a general description of the filtering problem, and form the basis of determining the states of objects. All subsequent filters are special cases derived from the Bayes filter given certain assumptions.

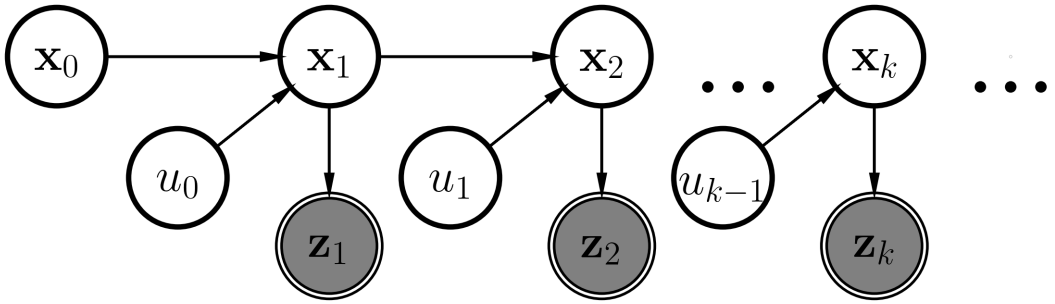


Figure 2.17: Bayesian network of the filtering problem for the discrete case. The nodes represent random variables with shaded nodes indicating measured random variables. The states $\mathbf{x}_k$ evolve over time and are measured at every discrete time step.

The Bayes filter is dependent on two conditional distributions. If the functions that describe these conditional distributions are known and specified, then all the distributions in the Bayes net are known and specified, with the exception of the initial state distribution $\mathbf{x}_0$. The derivation of the Bayes filter is described using these conditional distributions. The first is based on the *Markov assumption*, which states that the next hidden states, at $k+1$, are conditionally independent of the states at $k-1$, given the states at k and the inputs at k, expressed by $p(\mathbf{x}_k|\mathbf{x}_{k-1}, u_{k-1})$. $p(\mathbf{x}_k|\mathbf{x}_{k-1}, u_{k-1})$ denotes the distribution of the expected states given the previous state and robot input. This conditional behaviour is represented in Figure 2.17 with the lack of any casual links from, for example, $\mathbf{x}_0$ to $\mathbf{x}_2$.

The second conditional distribution is $p(\mathbf{z}_k|\mathbf{x}_k)$, which expressed the conditional distribution of $\mathbf{z}_k$ given $\mathbf{x}_k$, as illustrated in Figure 2.17 with the only causal link to $\mathbf{z}_k$ being from $\mathbf{x}_k$. $p(\mathbf{z}_k|\mathbf{x}_k)$ is the measurement model, and denotes the distribution of the expected measurements, given the states. If the functions that describe these conditional distributions as known, then the entire Bayes net is known.

The desired posterior distribution can be determined using Bayes' theorem for conditional densities, described by [68]

$$p(\mathbf{x}_k|\ \mathbf{z}_{0:k}, u_{0:k-1}) = \frac{p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\ \mathbf{z}_{0:k-1}, u_{0:k-1})}{p(\mathbf{z}_k|\ \mathbf{z}_{0:k-1}, u_{0:k-1})}. \tag{2.58}$$

The denominator term does not need to be explicitly determined since it functions as a normalisation factor for the resultant density function. In order to simplify the expression, the substitution described by

$$\frac{1}{\eta} = p(\mathbf{z}_k|\mathbf{z}_{0:k-1}, u_{0:k-1}) \tag{2.59}$$

is made, resulting in

$$p(\mathbf{x}_k|\mathbf{z}_{0:k}, u_{0:k-1}) = \eta\, p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{z}_{0:k-1}, u_{0:k-1}). \tag{2.60}$$

Substitutions described by

$$p^-(\mathbf{x}_k) = p(\mathbf{x}_k|\mathbf{z}_{0:k-1}, u_{0:k-1}) \tag{2.61}$$

and

$$p^+(\mathbf{x}_k) = p(\mathbf{x}_k|\mathbf{z}_{0:k}, u_{0:k-1}) \tag{2.62}$$

are also made to simplify the notation, where - and + denotes the distribution before and after the measurement update respectively. As a result, Equation 2.61 denotes the belief of $\mathbf{x}_k$ before being updated with measurements, and Equation 2.62 denotes the belief of $\mathbf{x}_k$ after having been updated with measurements.

The derivation results in the Bayes filter expressed by

$$p^-(\mathbf{x}_k) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1}, u_{k-1})p^+(\mathbf{x}_{k-1})d\mathbf{x}_{k-1} \tag{2.63}$$

and

$$p^+(\mathbf{x}_k) = \eta\, p(\mathbf{z}_k|\mathbf{x}_k)p^-(\mathbf{x}_k). \tag{2.64}$$

The derivation of Equation 2.63 is not expressed here, but is done by expressing Equation 2.61 as a marginal density function calculated from a joint distribution with conditional dependences.

The Bayes filter functions as a two stage-operation. Equation 2.63 is the state prediction step. The joint distribution of the next states of the system (at k) are inferred using prior knowledge of the system's states $p^+(\mathbf{x}_{k-1})$, given $p(\mathbf{x}_k|\mathbf{x}_{k-1},u_{k-1})$ that governs the evolution of the states. The second part is the measurement update stage described by Equation 2.64. Partial noisy observations, modelled by the measurement model $p(\mathbf{z}_k|\mathbf{x}_k)$, are used along with the result of the prediction step stage to yield the posterior distribution $p^+(\mathbf{x}_k)$. The filter function in a recursive manner, where the posterior at k becomes the prior at $k+1$. The filter can be implemented with Monte Carlo sampling as an approximation of the distributions [28].

The expected distribution of the next states given the previous states (the motion model) $p(\mathbf{x}_k|\mathbf{x}_{k-1},u_{k-1})$, can represented by the *transition density* notation $f_{k|k-1}(\mathbf{x}_k|\mathbf{x}_{k-1},u_{k-1})$ as

$$f_{k|k-1}(\mathbf{x}_k|\mathbf{x}_{k-1},u_{k-1}) \triangleq p(\mathbf{x}_k|\mathbf{x}_{k-1},u_{k-1}). \tag{2.65}$$

Equation 2.65 denotes the transition of the states from $k-1$ to k. Given the application of tracking a moving object in the environment, the states of a object would evolve according to dynamics such as a constant velocity or acceleration model. The expected distribution of the measurements given the states (the measurement model) $p(\mathbf{z}_k|\mathbf{x}_k)$ can represented by the *likelihood function* notation $g_k(\mathbf{z}_k|\mathbf{x}_k)$ as

$$g_k(\mathbf{z}_k|\mathbf{x}_k) \triangleq p(\mathbf{z}_k|\mathbf{x}_k) \tag{2.66}$$

Equation 2.66 denotes the transform of given states, in the state space, to measurements, in the measurement space. For the proposed application using stereo vision cameras as sensors, the camera projection transform derived in Section 2.1 models the likelihood function. Substituting the transition density (Equation 2.65) and likelihood function (Equation 2.66) into Equations 2.63 and 2.64 results in

$$p^-(\mathbf{x}_k) = \int f_{k|k-1}(\mathbf{x}_k|\mathbf{x}_{k-1},u_{k-1})p^+(\mathbf{x}_{k-1})d\mathbf{x}_{k-1} \tag{2.67}$$

and

$$p^+(\mathbf{x}_k) = \eta g_k(\mathbf{z}_k|\mathbf{x}_k)p^-(\mathbf{x}_k). \tag{2.68}$$

The Bayes filter results in estimates of the states of a single object with known measurement correspondence. It uses the transition density and measurement model of the relevant application to model the transforms present, and models the uncertainty present due to noise. Once features have been obtained and matched, they can be used in the Bayes filter framework to produce estimates of a moving object. This estimation process occurs recursively from discrete measurement to the next discrete measurement. However, the filter is computationally expensive, given a sampling implementation, and as a result other filters that are derived from the Bayes filter are more favoured for implementation.

2.4.1 Linear Kalman Filter

Given the computational complexity of the Bayes filter, assumptions are made in order to result in less expensive results. By making certain assumptions of the nature of the problem, the Bayes filter yields what is referred to as the *Kalman filter* (KF) equations. These equations can be derived by assuming, firstly, that the transition density and likelihood function are linear transforms. Secondly, that the noise present in the system, both process and measurement, are Gaussian (normally) distributed. The noise is assumed to be zero mean and independent, which also implies that it is uncorrelated given that it is assumed to be Gaussian distributed. If the initial distribution is assumed to be Gaussian distributed, then all subsequent state prediction and measurement update distributions would also be Gaussian distributed, given the linear transforms and uncorrelated noise assumptions. After these assumptions are made,

$$\mathbf{x}_k = \mathbf{F}_k \mathbf{x}_{k-1} + \mathbf{B}_k u_{k-1} + W_k \tag{2.69}$$

expresses the discrete state prediction with process noise. $\mathbf{F}_k$ is a matrix that denotes the linear motion model at k, $\mathbf{B}_k$ denotes how the control input influences the state, and W_k denotes the Gaussian distributed process noise defined by

$$W_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_k), \tag{2.70}$$

where $\mathcal{N}$ denotes a Gaussian distribution, where $\mathbf{Q}_k$ denotes the covariance of the distribution, and $\mathbf{0}$ the zero mean vector. The assumptions also result in a linear measurement model $\mathbf{H}_k$, from the likelihood function, described by

$$\mathbf{z}_k = \mathbf{H}_k \mathbf{x}_k + V_k, \tag{2.71}$$

where measurement noise V_k is defined by

$$V_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k). \tag{2.72}$$

Under these assumptions a Gaussian prior distribution $p^+_{k-1}(\mathbf{x}_{k-1}) \sim \mathcal{N}(\boldsymbol{\mu}^+_{k-1}, \Sigma^+_{k-1})$ results in a Gaussian posterior distribution $p^+_k(\mathbf{x}_k) \sim \mathcal{N}(\boldsymbol{\mu}^+_k, \Sigma^+_k)$. As a result, parameterised density-based descriptions along with simple linear equations can be used to obtain estimates. $p^-_k(\mathbf{x}_k) \sim \mathcal{N}(\boldsymbol{\mu}^-_k, \Sigma^-_k)$ denotes the resultant Gaussian distribution after the state predication and before the measurement update. The described linear system, with zero mean Gaussian distributed noise and Gaussian initial states, causes the Bayes filter to reduce to a set of simple equations expressed by

$$\boldsymbol{\mu}^-_k = \mathbf{F}_k \boldsymbol{\mu}^+_{k-1} + \mathbf{B}_k u_{k-1}, \tag{2.73}$$

and

$$\Sigma_k^- = \mathbf{Q}_k + \mathbf{F}_k \Sigma_{k-1}^+ \mathbf{F}_k^T, \tag{2.74}$$

for the state distribution propagation, and

$$\mathbf{L}_k = \Sigma_k^- \mathbf{H}_k^T (\mathbf{S}_k)^{-1} = \Sigma_k^- \mathbf{H}_k^T (\mathbf{R}_k + \mathbf{H}_k \Sigma_k^- \mathbf{H}_k^T)^{-1}, \tag{2.75}$$

$$\boldsymbol{\mu}_k^+ = \boldsymbol{\mu}_k^- + \mathbf{L}_k \widetilde{\mathbf{z}}_k = \boldsymbol{\mu}_k^- + \mathbf{L}_k (\mathbf{z}_k - \mathbf{H}_k \boldsymbol{\mu}_k^-), \tag{2.76}$$

and

$$\Sigma_k^+ = (I - \mathbf{L}_k \mathbf{H}_k) \Sigma_k^-, \tag{2.77}$$

the measurement update. I is the identity matrix, $\mathbf{L}_k$ represents the *Kalman gain* that updates the states given the predication and measurements. $\mathcal{N}(\mathbf{H}_k \boldsymbol{\mu}_k^-, \mathbf{S}_k)$ denotes the *innovation distribution* which is a Gaussian distribution with $\mathbf{S}_k$ the innovation covariance. The innovation denotes the distribution of the expected measurement in the measurement space. $\widetilde{\mathbf{z}}_k$ denotes the *reciprocal*, the difference between the obtained measurement and the expected measurement distribution mean. The Kalman gain operates as a weighting factor between the predicated states and the measurements. The more reliable the measurement (less measurement noise), the more emphasis the Kalman gain will put on the reliability of measurements in the update step, and vice versa.

The KF provides, relative to the Bayes filter, computationally cheaper density-based state estimates. The KF framework could be used to obtain state estimates of a moving object given the feature detection methods already presented, but there are two limitations. First, the issue of data correspondence, that is which measurements obtained with feature detection methods correspond to which states. This issue is addressed in Section 2.5. Second, the derived stereo vision projection transform in Section 2.1, which would function as the measurement model, is a nonlinear transform. The KF framework assumes linear motion and measurement models. As a result, either a Monte Carlo sampling approach [28] needs to be used, or an approximation in order to stay within the KF framework. An approximation is desirable given the computationally cheaper implementation and the convenience of parameter based densities.

2.4.2 Nonlinear Techniques

The KF provides simple operations on parameters of distributions, and yields estimates of the posterior distribution of a system's hidden states. These qualities are desirable, but are dependent on the linear dynamics assumption. In the event of nonlinear dynamics, a complex implementation in the Bayes filter framework could be used, or an approximation technique

could be used in order to stay within the KF framework. Staying within the KF framework is desirable given its simplicity and computational gain. It is also desirable given complex computational burden of a statistical sampling approaches such as Monte Carlo sampling. Two popular approximation techniques are *linearisation* and the *unscented transform*.

Extended Kalman Filter

Linearisation of dynamics in the KF framework is commonly referred to as the extended Kalman filter (EKF). The approximation linearises the nonlinear dynamics of the motion model $\underline{f}(\cdot)$ and measurement model $\underline{h}(\cdot)$, expressed by

$$\mathbf{x}_k = \underline{f}(\mathbf{x}_{k-1}, u_{k-1}) + W_k \tag{2.78}$$

and

$$\mathbf{z}_k = \underline{h}(\mathbf{x}_k) + V_k, \tag{2.79}$$

at the means of the relevant distributions, $\boldsymbol{\mu}_k^+$ and $\boldsymbol{\mu}_k^-$. A first-order Taylor series expansion of Equations 2.78 and 2.79 are used, expressed by the *Jacobian* of $f(\mathbf{x}_{k-1}, u_{k-1})$ and $h(\mathbf{x}_k)$, described by

$$\mathbf{F}_k = \left.\frac{\partial \underline{f}(\mathbf{x}, u_{k-1})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\boldsymbol{\mu}_{k-1}^+} \tag{2.80}$$

and

$$\mathbf{H}_k = \left.\frac{\partial \underline{h}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\boldsymbol{\mu}_k^-}, \tag{2.81}$$

where $\underline{f}(\cdot)$ and $\underline{h}(\cdot)$ are vector functions.

The first order Taylor series expansion results in

$$\mathbf{x}_k \approx \underline{f}(\boldsymbol{\mu}_{k-1}^+, u_{k-1}) + \mathbf{F}_k(\mathbf{x}_{k-1} - \boldsymbol{\mu}_{k-1}^+) + W_k \tag{2.82}$$

and

$$\mathbf{z}_k \approx \underline{h}(\boldsymbol{\mu}_k^-) + \mathbf{H}_k(\mathbf{x}_k - \boldsymbol{\mu}_k^-) + V_k. \tag{2.83}$$

These equations express the function values at the linearisation points, for example, $\underline{h}(\boldsymbol{\mu}_k^-)$, and the first order terms, for example $\mathbf{H}_k(\mathbf{x}_k - \boldsymbol{\mu}_k^-)$, while ignoring the higher order terms.

These linearised dynamics, expressed by Equations 2.80 and 2.81, used in the KF framework constitutes the EKF. The dynamics are iteratively linearised around the latest known estimate mean of the states at every discrete time instant using the Jacobian. The EKF is usable, but is not as desirable as the unscented transform approach.

Unscented Kalman Filter

The unscented transform is an alternative method to the EKF Taylor approximation. The unscented transform is a deterministic sampling approach used for handling nonlinearities. This method used in a Kalman filter framework is referred to as the unscented Kalman filter (UKF). The unscented transform draws samples from the prior Gaussian distribution called *sigma points*, with the understanding that the sigma points provide a suitable degree of representation of the sampled distribution. The relevant nonlinear transforms are applied to the sigma points, and a Gaussian distribution is fitted to the resulting transformed points.

Define X as a jointly distributed multi-variant Gaussian distribution with n dimensions, described by

$$X \sim \mathcal{N}(\boldsymbol{\mu}_X, \Sigma_X). \tag{2.84}$$

An arbitrary nonlinear transform (motion or measurement model) is applied to X to create Y as expressed by

$$Y = \underline{f}(X) = \begin{bmatrix} f_1(X) \\ \vdots \\ f_n(X) \end{bmatrix}. \tag{2.85}$$

The unscented transform draws $2n+1$ sigma points $\chi^{[i]}$ from X as described by [51]

$$\chi^{[0]} = \boldsymbol{\mu}_X, \tag{2.86}$$

$$\chi^{[i]} = \boldsymbol{\mu}_X + \left(\sqrt{(n+\lambda)\Sigma_X}\right)_i, \ i = 1, \ldots, n, \tag{2.87}$$

and

$$\chi^{[i]} = \boldsymbol{\mu}_X - \left(\sqrt{(n+\lambda)\Sigma_X}\right)_i, \ i = n+1, \ldots, 2n. \tag{2.88}$$

Two sigma points are drawn for each dimension along with $\chi^{[0]}$, selected as the mean of the distribution. $(\cdot)_i$ denotes the i-th column of a matrix. The square root of a matrix ($B = \sqrt{A}$) is a matrix that satisfies $A = BB^T$. α and κ are parameters that determine the spread of the sigma points around the mean as

$$\lambda = \alpha^2(n+\kappa) - n. \tag{2.89}$$

Next weighting factors are calculated that are used when calculating the mean ($\boldsymbol{\mu}_Y$) and covariance (Σ_Y) of the Gaussian approximation of the distribution of Y that will be fitted to the

transformed sigma points $\mathcal{Y} = \underline{f}(\chi)$. β is usually chosen to be 2, and κ to be 0. These weights are expressed as

$$w_m^{[0]} = \frac{\lambda}{n+\lambda}, \tag{2.90}$$

$$w_c^{[0]} = \frac{\lambda}{n+\lambda} + (1-\alpha^2+\beta), \tag{2.91}$$

and

$$w_m^{[i]} = w_c^{[i]} = \frac{1}{2(n+\lambda)}, \ i = 1,\ldots,2n. \tag{2.92}$$

The mean and covariance of Y can now be calculated using

$$\boldsymbol{\mu}_Y \approx \sum_{i=0}^{2n} w_m^{[i]} \mathcal{Y}^{[i]} \tag{2.93}$$

and

$$\Sigma_Y \approx \sum_{i=0}^{2n} w_c^{[i]} (\mathcal{Y}^{[i]} - \boldsymbol{\mu}_Y)(\mathcal{Y}^{[i]} - \boldsymbol{\mu}_Y)^T. \tag{2.94}$$

While the EKF only subjects the means of the distribution estimates to the nonlinear transforms, and linearises at the means, the UKF subjects all the drawn sigma points to the nonlinear dynamics. As a result, the UKF results in the same or better accuracy than the EKF, depending on the degree of the nonlinear dynamics [51], [26]. This is due to the fact that the nonlinear dynamics are conserved and used for higher orders of representation of the distributions, while neglected for the EKF.

The presented filter, Bayes filter, KF, EKF, and UKF, are all capable of estimating the states of a moving object in a robot's environment. These filters' formulation includes the certainty with which states are known and account for the noise present in the system which make state predictions and measurements less reliable. These filters can also be used for multiple time-varying objects, assuming that the origin of each measurement is known, that is which state caused which measurement. However, the assumption is not valid for vision-based applications and needs to be accounted for in order to known which extracted features should be associated with which states.

2.5 Multi-Target Tracking and Data Association

In most real-world state estimation problems there are often multiple targets (objects) that are observed at every measurement instant. Note that tracking a target and tracking an object

is synonymous, data association literature tends to use 'target' terminology while DATMO uses 'object'. As a result, the problem definition switches to the estimation and tracking of the trajectories of a unknown time-varying number of targets with uncertain measurement-to-target associations in the presence of measurement clutter. Measurement clutter is used to refer to obtained measurements that are not generated by any of the targets currently present. This problem is known as multi-target tracking (MTT). This new problem definition causes an issue in a single target probabilistic filtering context, since knowledge about which target generated which measurement is not known. The single target KF is able to estimate the joint distribution of the states of a single target in the presence of measurement and process noise. However, in order for the Bayesian filtering framework approach to be feasible for the MTT problem, the exact origins of each measurement would need to be known. This problem is known as the *unknown correspondence* problem or *data association* problem, that is which target caused which measurement.

Updating the state estimates of a target with an inappropriate measurement will cause incorrect state estimates. That is associating a measurement with an incorrect target. The errors will be present in the state estimates of the target for a while given that the KF framework assumes perfect measurement-to-target correspondence. The KF framework also does not smooth out previous estimates, that is the KF only conducts inference forward in time, not back. A strategy for associating measurements to their correct corresponding targets is required before a Bayesian filtering framework, and all of its desirable characteristics, can be used.

Three popular and influential strategies for addressing the data association issue are called global nearest neighbour (GNN), joint probabilistic data association (JPDA), and multiple hypothesis tracking (MHT). These strategies each address the data association problem with a unique approach [19]. The GNN approach retains the KF framework, and makes a hard association with the most likely correct association. JPDA also retains the KF framework, but generates a pseudo-measurement that is a weighted combination of multiple measurements. MTT makes every possible association, also within the KF framework. The following subsections will investigate and expound how these approach solve the unknown data association problem in more detail. These techniques are necessary for a robot to reliably track moving objects in its environment.

2.5.1 Global Nearest Neighbour

Global nearest neighbour (GNN) is the simplest solution of the methods listed. GNN operates on the principle of considering which measurement-to-target association hypothesis is the most likely to be true with each measurement update [24]. The chosen association hypothesis is subsequently used to update the state estimates of the relevant target. This strategy is expanded to include multiple targets. GNN makes a hard association between a measurement

and underlying target by inspecting which measurement is the closest to the expected measurement (in the statistical sense), which is the most likely association hypothesis, by means of statistical distances [24].

GNN uses a *gating strategy* to eliminate highly unlikely data association hypotheses. Only measurements within the *validation gate* of a target are considered and inspected. A validation gate is a region in the measurement space centred at the mean of the Gaussian innovation distribution $\mathcal{N}(\mathbf{H}_k\boldsymbol{\mu}_k^-, \mathbf{S}_k)$, with bounds proportional to a standard deviation confidence bounds of the innovation covariance. Conceptually, a gate can be thought of as a confidence bounds contour of the innovation distribution, at whatever level of desirable uncertainty. This strategy assumes that the desired measurement will be within the vicinity of the validation gate, which is a reasonable assumption given that the certainty of the state estimate is known and that the likelihood function and transition density are known.

Mahalanobis Distance

The *Mahalanobis distance* D_{MAL} metric, described by

$$D_{MAL} = \sqrt{(\mathbf{x}-\boldsymbol{\mu})^T(\Sigma)^{-1}(\mathbf{x}-\boldsymbol{\mu})}, \tag{2.95}$$

is used to determine whether a measurement is within the validation gate. The Mahalanobis distance metric is a mean and covariance normalised distance metric of a sample point $\mathbf{x}$ with respect to a Gaussian distribution $\mathcal{N}(\boldsymbol{\mu},\Sigma)$. The distribution is normalised with respect to the mean and variance, and as a result the distance is that of the L2 norm of a standard normal distribution $\mathcal{N}(\mathbf{0},I)$, with I denoting the identity matrix. This metric can be thought of as an indication of statistical deviation, that is how many standard deviations are the sample point $\mathbf{x}$ from the mean $\boldsymbol{\mu}$. The metric is a scalar value and the distribution space can be multidimensional. Measurements found to be within a validation gate are deemed *validated measurements*. The Mahalanobis distance of measurements with respect a Gaussian validation gate is used to determine which measurements are validated.

The distances between all measurements within the validation gate and the mean of the gate are inspected and the most likely hypothesis is chosen, that is the smallest distance. Subsequently, a hard data association is made between the relevant target and the chosen measurement. In the situation depicted in Figure 2.18, the distances between the red track and the elements in the measurement set $\{\mathbf{z}_1, \mathbf{z}_3, \mathbf{z}_6, \mathbf{z}_8\}$ are used to determine the most likely hypothesis. Similarly, the distances between the blue track and the elements in the measurement set $\{\mathbf{z}_1, \mathbf{z}_4, \mathbf{z}_6\}$ are used to determine the most likely hypothesis. An assumption is made that every target only generates one measurement [24]. As a result, measurements that have been associated with a target are not considered as potential hypotheses for other targets. That is they are removed

from other subsets. In the event that a given measurement is the most likely hypothesis for two or more targets, then the most likely association is made, that is the shortest distance. The single hard association is used in a KF framework in the same way that a known correspondence for a single target would be used. GNN runs the risk of associating a clutter measurement with a target, and as a result, producing incorrect state estimates.

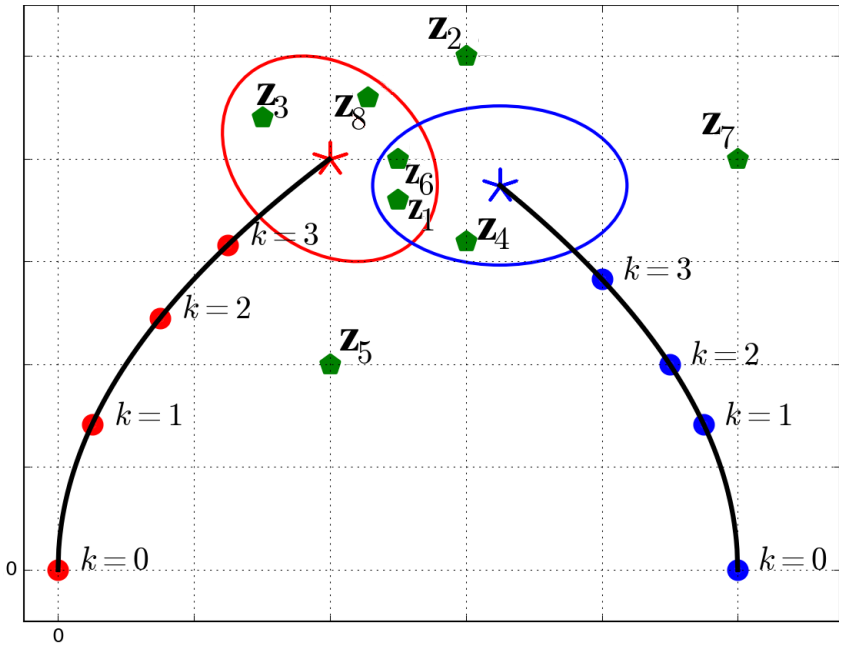


Figure 2.18: Two dimensional illustration of validation gates and measurements in the measurement space. Two tracks, red and blue, have validation gates at $k = 4$ with 8 obtained measurements, indicated by green. Set $\{\mathbf{z}_1, \mathbf{z}_3, \mathbf{z}_6, \mathbf{z}_8\}$ is within the the red track's validation gate. Set $\{\mathbf{z}_1, \mathbf{z}_4, \mathbf{z}_6\}$ is within the the blue track's validation gate.

2.5.2 Probabilistic Data Association

Probabilistic data association (PDA) is a suboptimal soft assignment association strategy, as opposed to GNN which is a hard assignment association strategy [19]. Instead of making one hard association, a pseudo-measurement is generated and subsequently associated with the target [20]. PDA assumes a single target in the environment, and has been extended to a multi-target variant.

PDA generates a pseudo-measurement that is a function of all the measurements within the validation region of the target. These validated measurements are all added together, each with their own weighting factor. The weights are a function of the association hypothesis likelihood, that is validated measurements which are more likely to have been generated by the target have larger weights, and consequently contributed more to the generation of the

pseudo-measurement. The linear Kalman filter Gaussian mean update equation is expressed by

$$\boldsymbol{\mu}_k^+ = \boldsymbol{\mu}_k^- + \mathbf{L}_k \widetilde{\mathbf{z}}_k. \tag{2.96}$$

The KF filter puts appropriate weight on the measurement given the process and measurement noise with the Kalman gain ($\mathbf{L}_k$). The reciprocal $\widetilde{\mathbf{z}}_k$, expressed as

$$\widetilde{\mathbf{z}}_k = \mathbf{z}_k - \mathbf{H}_k \boldsymbol{\mu}_k^-, \tag{2.97}$$

is the difference between the obtained measurement and the expected measurement mean. In the PDA framework, this reciprocal is constructed with all validated measurements as described by

$$\widetilde{\mathbf{z}}_k = \sum_{j=1}^{m_{val}} q_j \widetilde{\mathbf{z}}_k^{(j)} = \sum_{j=1}^{m_{val}} q_j \left(\mathbf{z}_k^{(j)} - \mathbf{H}_k \boldsymbol{\mu}_{k|k-1} \right), \tag{2.98}$$

where m_{val} is the number of validated measurements, $\widetilde{\mathbf{z}}_k^{(j)}$ is the j-th validated measurement reciprocal. q_j is used as the weight for each validated measurement. q_j is the *posterior association probability* that measurement j was generated by the target (event χ_j) described by

$$q_j = p(\chi_j | \, Z_k^{all}), \quad j = 0, 1, 2, ..., m_{val}, \tag{2.99}$$

where Z_k^{all} is the collection of all validated measurement over all time $Z_k^{all} = Z_{k-1}^{all} \cup \{\mathbf{z}_k^{(1)}, ..., \mathbf{z}_k^{(m_{val})}\}$. $j = 0$ is the notation used to indicate no association, the probability that the measurement was generated by clutter. The greater the likelihood that $\mathbf{z}_k^{(j)}$ was generated by the target, the larger its association probability q_j will be, and hence the larger its contribution to the weighted pseudo-measurement will be.

The PDA filter framework results in

$$\boldsymbol{\mu}_k^+ = \boldsymbol{\mu}_k^- + \mathbf{L}_k \sum_{j=1}^{m_{val}} q_j \widetilde{\mathbf{z}}_k^{(j)}, \tag{2.100}$$

$$\Sigma_k^+ = \Sigma_k^- - \left[(1 - q_0) \mathbf{L}_k \mathbf{H}_k \Sigma_k^- \right] + \Sigma_c, \tag{2.101}$$

and

$$\Sigma_c = \mathbf{L}_k \left[\sum_{j=1}^{m_{val}} q_j \widetilde{\mathbf{z}}_k^{(j)} (\widetilde{\mathbf{z}}_k^{(j)})^T - \widetilde{\mathbf{z}}_k (\widetilde{\mathbf{z}}_k)^T \right] \mathbf{L}_k^T, \tag{2.102}$$

update equations. These are the augmented variants of the standard linear KF update equations. $\widetilde{\mathbf{z}}_k^{(j)}$ is the j-th reciprocal and $\widetilde{\mathbf{z}}_k$ is defined by Equation 2.98. Σ_c changes the covariance in order to account for the spread of the individual measurements used. The probability of association term $(1-q_0)$ (since q_0 is the probability of no association, that is clutter generated) adjusts the covariance measurement update component. The association probabilities can be calculated with [31]

$$q_j = \frac{\exp(-0.5(\widetilde{\mathbf{z}}_k^{(j)})^T \mathbf{S}_k^{-1} \widetilde{\mathbf{z}}_k^{(j)})}{b + \sum_{j=1}^{m_{val}} \exp(-0.5(\widetilde{\mathbf{z}}_k^{(j)})^T \mathbf{S}_k^{-1} \widetilde{\mathbf{z}}_k^{(j)})} \quad j = 1,2,...,m_{val} \tag{2.103}$$

and

$$q_0 = \frac{b}{b + \sum_{j=1}^{m_{val}} \exp(-0.5(\widetilde{\mathbf{z}}_k^{(j)})^T \mathbf{S}_k^{-1} \widetilde{\mathbf{z}}_k^{(j)})}, \tag{2.104}$$

with

$$b = (2\pi)^{m_{dim}/2} c |\mathbf{S}_k|^{0.5} (1 - p_D p_G)/p_D = (2\pi)^{m_{dim}/2} \frac{c v_{gate}}{c_M \sigma^{m_{dim}}} (1 - p_D p_G)/p_D \tag{2.105}$$

and

$$v_{gate} = c_M \sigma^{m_{dim}} |\mathbf{S}|^{0.5}. \tag{2.106}$$

q_0 denotes no association and hence expresses the probability that the measurement was generated by clutter. Clutter measurements are assumed to be Poisson distributed and the correct measurement is assumed to be Gaussian distributed. v_{gate} is the volume of the surveillance region, the validation gate. c is the expected number of clutter measurements per unit volume (number/volume, clutter density), p_D is the probability of detection, and p_G is the probability that the generated measurement is within the surveillance region assuming it was detected. m_{dim} is the dimensional order of the measurement space, c_M is the volume of a unit sphere (depending on the dimensional order of measurement space), and σ (standard deviations) is the gating threshold used.

PDA is suboptimal (in comparison with a correct hard association) since the correct measurement is consequently associated with the track, but the measurement is merged together with other measurements not generated by the target. As a result, a suboptimal estimate is generated but the risk of making an incorrect association is largely avoided. PDA avoids the risk of making an incorrect association, the biggest risk of GNN, but the trade-off is a suboptimal estimate. PDA is quite desirable in filtering contexts characterised by the presence of a large amount of clutter. PDA assumes a single target in the environment, but multiple PDA filters

could be used simultaneously in parallel for multiple targets. However, this approach causes PDA to suffer from the *process of coalescence* [42]. The process describes a phenomenon where the track estimates of two or more targets are drawn together. The PDA framework does not account for multiple targets, and as a result assumes that every measurement is either generated by the target or by clutter. Therefore, the filter does not take the likelihood that a given measurement could have been generated by another target into account, which results in coalescence. This problem is addressed with JPDA.

2.5.3 Joint Probabilistic Data Association

Coalescence occurs, the PDA framework, when at least one obtained measurement appears within the validation gates of two or more targets. This shared measurement pulls the pseudo-measurements used in the update stage of the relevant targets toward each other, which gradually draws their tracks together. PDA does not take the relative probability that certain measurements could have originated from other targets into account, but Fortmann [31] resolved this issue with joint probabilistic data association (JPDA).

JPDA is the multi-target variant of PDA and mitigates the coalescence issue by incorporating the probability that measurements could be generated by one of many targets or clutter [17]. JPDA uses similar principles to the PDA approach, but calculates its association weights using a joint probabilistic score, which incorporates the relationship between all measurement association combinations with all targets. The derivation is reproduced from Fortmann's JPDA paper [31].

The conditional probabilities of events described by

$$\chi = \bigcap_{j=1}^{m_{val}} \chi_{jt_j} \tag{2.107}$$

are now jointly determined. χ_{jt_j} denotes the event that measurement j was generated by target t_j. t_j denotes the index of the target that caused measurement j. Index $t_j = 0$ indicates that measurement j was generated by clutter (target 0). The JPDA framework makes the assumption that a given measurement can only be caused by a single target, that is for $j \neq l$ and $t_j > 0$ (a non-clutter generated measurement) results in $t_j \neq t_l$ (indices can't be the same). However, the target indices can be the same if measurements were generated by clutter, that is multiple measurements can be generated by clutter. These assumed conditions determine the *feasible joint events* that need to be determined as described by Equation 2.107 [31].

The feasible joint association events can be represented by a *validation matrix* $\Omega = [\omega_{jt}]$ which consists of binary elements indicating whether an association is feasible or not. j indicates a

given measurement and t a given target. Feasible associations are determined by the assumptions that a measurement can only be caused by a single target, and also measurements need to be validated in order to be considered [31]. From the validation matrix each individual feasible event χ can be represented by a matrix $\hat{\Omega}(\chi) = [\hat{\omega}_{jt}(\chi)]$, with $\hat{\omega}_{jt}(\chi)$ defined by

$$\hat{\omega}_{jt}(\chi) = \begin{cases} 1, & \text{if } \chi_{jt} \text{ is realised} \\ 0, & \text{if } \chi_{jt} \text{ is not realised} \end{cases} . \tag{2.108}$$

The footnote of t is dropped to indicate a generic target and not necessarily the correct target, that is the target that caused the measurement.

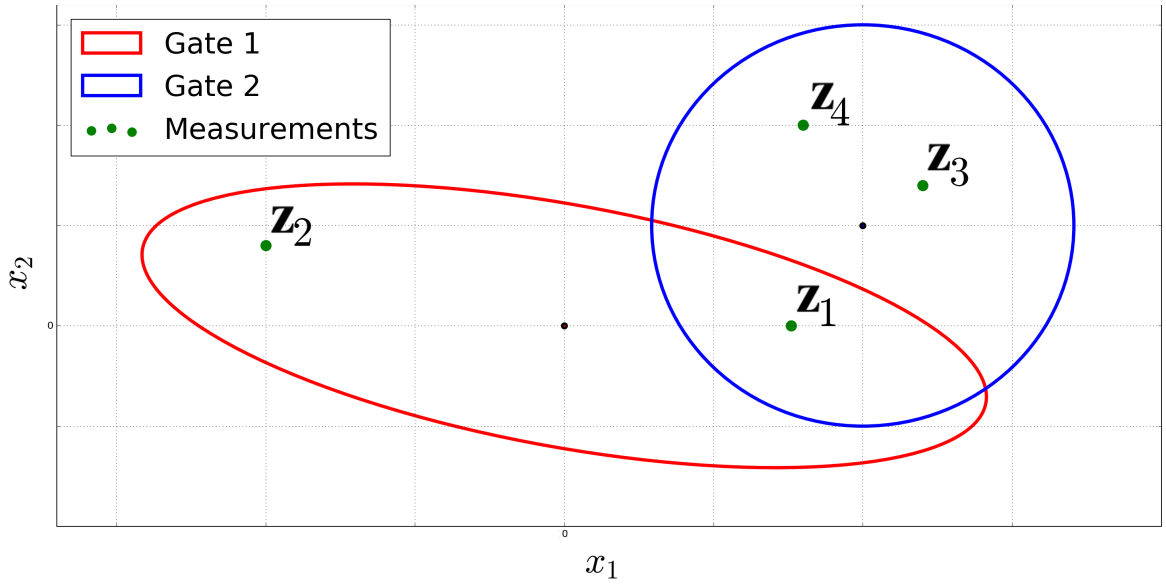


Figure 2.19: JPDA example. Red (R) and blue (B) confidence ellipses represent the gates, green represents the measurements.

The validation matrix for the example depicted by Figure 2.19 is expressed by

$$\Omega = \overbrace{\begin{matrix} 0 & R & B \end{matrix}}^{t} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix} \left.\begin{matrix} 1 \\ 2 \\ 3 \\ 4 \end{matrix}\right\} j. \tag{2.109}$$

$\omega_{jt} = 1$ indicates a validated association. The first row of the matrix indicates that measurement $j = 1$ could have originated from either the red or blue target. The first column is

populated by ones in order to indicate that all the measurements could have originated from clutter. The validation matrix can now be used to represent each individual possible combination of association events $\chi_{jt} = 1, 2, ..., e_n$ where e_n is the total number of feasible association combinations. Equations 2.110 are all the feasible events that could be realised from Equation 2.109 as depicted by Figure 2.19 [22]. Each column $t \neq 0$ is only allowed to have a single one, and each row is only allowed to have a single one. These constrains emerge from the assumptions that a measurement is only allowed to be caused by a single target and that each target is only allowed to cause a single measurement, expect the clutter which is allowed to cause multiple measurements.

$$
\begin{aligned}
\hat{\Omega}(1) &= \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \quad \hat{\Omega}(2) = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \quad \hat{\Omega}(3) = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \\
\hat{\Omega}(4) &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \quad \hat{\Omega}(5) = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \quad \hat{\Omega}(6) = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \\
\hat{\Omega}(7) &= \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \quad \hat{\Omega}(8) = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \hat{\Omega}(9) = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \\
\hat{\Omega}(10) &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \quad \hat{\Omega}(11) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}
\end{aligned}
\tag{2.110}
$$

All target state estimate updates are done using the same form provided by the PDA framework described by equations 2.100 through 2.102, but with the posterior association probability, the probability that measurement j was caused by target t, described by [31]

$$
q_j^t = \sum_{\chi} p(\chi | Z_k^{all}) \hat{\omega}_{jt}(\chi), \tag{2.111}
$$

$$q_0^t = 1 - \sum_{j=1}^{m_{val}} q_j^t; \quad t = 0, 1, ..., T; \quad j = 1, ..., m_{val}, \tag{2.112}$$

and

$$p(\chi|Z_k) = \frac{c^{\phi(\chi)}}{c_{norm}} \prod_{j:\tau_j(\chi)=1} \frac{\exp[-0.5(\widetilde{\mathbf{z}}_{t_j}^{(j)})^T \mathbf{S}_{t_j}^{-1} (\widetilde{\mathbf{z}}_{t_j}^{(j)})]}{(2\pi)^{m_{dim}/2} \, |\mathbf{S}_{t_j}|^{0.5}} \prod_{t:\delta_t(\chi)=1} p_D^{(t)} \prod_{t:\delta_t(\chi)=0} (1 - p_D^{(t)}). \tag{2.113}$$

Equation 2.111 states that in order to calculate the probability that measurement j was caused by target t, q_j^t, we sum over all of the association combination probabilities for which the association is feasible, that is where the event occurs. T denotes the total number of targets. The k subscript has been dropped from $\widetilde{\mathbf{z}}$ for brevity's sake.

The clutter is assumed to be uniformly distributed with density v_{all}^{-1} (volume of surveillance region), measurements are assumed to be Gaussian distributed, and false measurements are assumed to be Poisson distributed, similar to the PDA framework. c is the expected number of clutter measurements per unit volume. $\delta_t(\chi)$ is the target detection indicator, a binary value that indicates whether any measurement is associated with a given target in event χ. $\tau_j(\chi)$ is the measurement association indicator, a binary value that indicates whether any target is associated with a given measurement in event χ. $\phi(\chi)$ is the number of false measurements in event χ. $p_D^{(t)}$ is the probability of detection for target t, $\mathbf{S}_{t_j}$ is the innovation covariance for target t_j. $\widetilde{\mathbf{z}}_{t_j}^{(j)}$ is the reciprocal of measurement j with respect to target t_j. m_{dim} is the dimensionality of the measurement space. c_{norm} is the normalisation factor required for the result to be a valid density. It can be determined by summing all the numerators for each χ.

JPDA accounts for the likelihoods that measurements could have been generated by one of many targets or clutter. This accommodation solves the issue of coalescence associated with multiple PDA filers working in parallel, which do not explicitly account for the presence of multiple targets. JPDA still avoids the incorrect hard association risk associated with GNN, and is also suboptimal just like PDA, but resolves the biggest issue associated with PDA. However, JPDA suffers from combinatorial complexity [36], where, even with the feasible event assumptions expressed in this subsection, the complexity of all the possible combinations of different associations becomes too much for implementation. If the number of targets and measurements are low, then the approach is suitable, however given the amount of measurements associated with feature detection, the approach becomes impractical due to the complexity.

2.5.4 Multiple Hypothesis Tracking

Multi-hypothesis tracking (MHT) is the alternative to both GNN and JPDA to solve the problem of data association in a MTT framework such as DATMO. The aim of MHT is to make all possible data association hypotheses, thereby determining the target estimates for every association hypothesis, instead of making a hard data association of the most likely hypothesis [24]. All measurements are separately associated with the target in question. All newly updated estimates are subsequently propagated and the process repeats. This approach is computationally expensive in comparison with GNN, but ensures that all the correct associations have indeed been made, even though they are mixed in with a multitude of incorrect associations and clutter associations. MHT can be thought of as multiple hard associations.

MHT gives way to an unbounded amount of different target tracks, and needs to be mitigated in order to be practically implemented [59]. Some implementations determine the association probability of each hypothesis and discards hypotheses that grow increasing unlikely over time. This discarding process limits the amount of hypotheses and avoids the unbounded track number problem.

MHT is of interest for the purpose of robust and reliable DATMO. MHT does not suffer from the risk of an incorrect single association present in the GNN framework, nor from coalescence in the PDA framework. MHT does not result in a suboptimal result such as present in the PDA and JPDA frameworks due to the weighted pseudo-measurements, but in an optimal result. MHT, although potentially expensive, does not suffer from combinatorial complexity such as JPDA. The framework is especially of interest for the design goals of robustness and reliability, which is due to the exhaustive association strategy that ensures that the correct association is made. Section 2.6 is dedicated to the derivation of a MHT approach in a Bayesian framework.

2.6 Multiple Hypothesis Tracking with Bayesian Statistics

The *multi-target Bayes filter* was derived to be the multi-target variant, and generalised expression, of the single-target Bayes filter. The multi-target Bayes filter falls within the domain of MHT, and tracks multiple data association hypotheses using a Bayes statistical formulation. This filter is desirable for DATMO with cameras given the data association problem present, and the advantageous afforded by a Bayesian framework to model uncertainty. The approach also deals with the issue of an unknown number of targets, given that the number of targets are not only unknown, but are time-varying.

The derivation is made possible by expressing the problem in a random set filtering formulation by modelling states and measurements as *random finite sets* (RFSs). A RFS is a set

containing a finite number of elements, and is characterised by a distribution over the cardinality of the set, that is there is uncertainty in the number of elements in the set. The set's elements are jointly distributed, given that a specific cardinality has been realised [48]. With the stated definition, a RFS can be thought of as random variable of which the value is a set of unknown number of elements [52]. The cardinality distribution is typically modelled as a discrete distribution since the number of elements in the set has to be an integer.

Random Finite Sets

This subsection focuses on the RFS formulation and its role in the larger derivation. The RFS formulation is ideal for the multi-target filtering problem since the dynamically varying number of targets in the presence of missed detections and clutter are accommodated by the RFS formulation [48]. Clutter being measurements obtained that were not generated by targets, and missed detections being a scenario where no measurements of any given target was obtained. All time varying number of estimates are simply added to the same set, while all target generated measurements and unwanted clutter are simply added to the same set [52].

This formulation allows for the filtering problem to express itself in an analogous manner to the Bayes filter. The analogy being that, in the Bayes filter, target state estimates are filtered in the single-target state space, whereas with the multi-target Bayes filter, target set state estimates are filtered in the multi-target state space. The analogy also extends to the degree that uncertainty in the single-target state space is characterised by a random variable, while uncertainty in the multi-target state space is characterised by a RFS. RFS formulation avoids the traditional data association problem since the order in which the individual elements appear in a given set is irrelevant [48].

The individual target states and measurements at a given time instance k are modelled using

$$X_k = \{\mathbf{x}_k^{(1)}, ..., \mathbf{x}_k^{(n(k))}\}, \;\; X_k \in \chi_{space}^{Mul} \tag{2.114}$$

and

$$Z_k = \{\mathbf{z}_k^{(1)}, ..., \mathbf{z}_k^{(m(k))}\}, \;\; Z_k \in \mathcal{Z}_{space}^{Mul} \tag{2.115}$$

respectively. Given $n(k)$ target states and $m(k)$ measurements at k, the respective states and measurements groupings are added together in sets X_k and Z_k on the multi-target state space χ_{space}^{Mul} and the multi-target observation space $\mathcal{Z}_{space}^{Mul}$ respectively. The multi-target state space is the collection all the subsets containing elements on the single-target state space, $\mathbf{x}_k^{(i)} \in \chi_{space}$. Similarly, the multi-target observation space is the collection all the subsets containing elements on the single-target observation space, $\mathbf{z}_k^{(i)} \in \mathcal{Z}_{space}$. RFS formulation of states and measurements also hold for the null (empty) set $\emptyset$.

The time evolution of the RFS of states from $k-1$ to k is defined by

$$X_k = \left[\bigcup_{\varsigma \in X_{k-1}} P_{k|k-1}(\varsigma)\right] \cup \left[\bigcup_{\varsigma \in X_{k-1}} B_{k|k-1}(\varsigma)\right] \cup \Gamma_k, \tag{2.116}$$

where $\varsigma \in X_{k-1}$. This evolution is characterised by three types of dynamics, namely time evolution of existing target states, spawned targets from existing target states, and newly birthed targets [48]. The resulting set is the union of these individual sets. $p_{k|k-1}(\cdot)$ is a RFS containing the results of propagating the existing target states over time. $B_{k|k-1}(\cdot)$ is a RFS containing all spawned targets. Spawned targets are targets that are created from existing targets and thus are state depended as represented in Equation 2.116. Γ_k represents the RFS of all randomly birthed targets in the state space. The measurement RFS at k is described by

$$Z_k = K_k \cup \left[\bigcup_{x \in X_k} \Theta_k(\mathbf{x})\right]. \tag{2.117}$$

The set consists of the union of the RFS of clutter measurements K_k, not generated by targets in the observed environment, and the RFS of measurements $\Theta_k(\cdot)$ that were generated by targets.

2.6.1 Multi-target Bayes filter

Given the RFS formulation, and subsequent multi-target state evolution and multi-target observation (Equations 2.116, 2.117), expressions for the multi-target transition density $f_{k|k-1}(X_k|X_{k-1})$ and multi-target likelihood functions $g_k(Z_k|X_k)$ can be derived using finite set statistics [48], [46]. Both $f_{k|k-1}(\cdot|\cdot)$ and $g_k(\cdot|\cdot)$ are traditionally used in literature to indicate both the single and multi-variant transition density and likelihood functions respectively, even though the single-target variants of the operations operate on the single-target state space, while the multi-target variants of the operations operate on the multi-target state space [48], [16]. The prediction-update steps of the multi-target Bayes filter are described by [48]

$$p_{k|k-1}(X_k|Z_{1:k-1}) = \int f_{k|k-1}(X_k|X_{k-1}) p_{k-1}(X_{k-1}|Z_{1:k-1}) \mu_s(dX_{k-1}) \tag{2.118}$$

and

$$p_k(X_k|Z_{1:k}) = \frac{g_k(Z_k|X_k) p_{k|k-1}(X_k|Z_{1:k-1})}{\int g_k(Z_k|X_{k-1}) p_{k|k-1}(X_{k-1}|Z_{1:k-1}) \mu_s(dX_{k-1})}. \tag{2.119}$$

μ_s denotes a reference measurement and normalising factor on the multi-target state space χ^{Mul}_{space} [48]. $p_k(X_k|Z_{1:k})$ represents the multi-target posterior distribution, $p_{k-1}(X_{k-1}|Z_{1:k-1})$ represents the multi-target prior distribution and $Z_{1:k}$ represents the collection of all measurement sets up until the current time instance k. $p_{k|k-1}(X_k|Z_{1:k-1})$ denotes the state inference

(state prediction) multi-target distribution predicted from $k-1$ to k $(k|k-1)$. Note that the RFS distribution notation $(k|k-1)$ differs from the random variable distribution notation ($+$ and $-$). The multi-target Bayes filter prediction-update steps are clearly analogues to the single-target variant described by Equations 2.63 and 2.64 with integrals on the joint multi-target state space, instead of the single-target state space, and with RFSs modelling the uncertainty instead of random variables.

2.6.2 Probability Hypothesis Density Filter

The multi-target Bayes filter is computationally intractable due to integrals on the multi-target state space χ_{space}^{Mul} over multi-target distributions. This practical limitation has largely been overcome by using a first-order statistical moment of the RFS distributions. The first-order statistical moment is the *expected value* of the RFS distribution [46], [48], known as the *probability hypothesis density* (PHD) or *intensity* [47], denoted by $v(\cdot)$ for the multi-target posterior and prior distributions. For RFS X on the multi-target state space χ_{space}^{Mul}, with distribution $p(X)$, the intensity (PHD) $v(\cdot)$ is a nonnegative function such that for a given region $S_R \subseteq \chi_{space}^{Mul}$ the expected number of elements in RFS X within the region S_R is characterised by

$$\int |X \cap S_R| p(X)(dX) = \int_{S_R} v(\mathbf{x}) d\mathbf{x} = \mathbb{E}[|X \cap S_R|]. \tag{2.120}$$

$|\cdot|$ of a set denotes the cardinality of the set. Equation 2.120 states that the expected number of elements in the RFS X can be determined by integrating over the PHD of X [48], assuming S_R is sufficiently large to encompass all set items, or is infinitely large and therefore negligible, given that S_R functions by reducing the multi-target state space dimensional limits. Given that $p(X)$ is a valid distribution, integrating over the entire domain (S_R is infinitely large, negligible) would result in $|X|$, the number of elements in the set.

The *PHD filter* can be derived from the multi-target Bayes filter by using the PHD (expectation) of all RFSs and by making the following three assumptions [52]:

1) Targets dynamically evolve in the state space and generate measurements in the measurement space completely independently, that is with no cross-coupled dynamics between targets.

2) The clutter subset of the measurement set is characterized by a Poisson random finite set and is independent of the rest of the measurement set, that is the measurements generated by the targets in the state space.

3) The multi-target RFS $p_{k|k-1}(X_k|Z_{1:k-1})$ obtained as a result of propagating the multi-target states via the likelihood function is characterized by a Poisson RFS.

Just as Poisson distributions on the single-target state space are completely characterised by their expectation, so too Poisson RFSs are completely characterised by their expectation, that is their intensity (PHD). A Poisson RFS's cardinality distribution is distributed according to a Poisson distribution, and for a given cardinality (evaluated PHD) all elements are independently distributed.

Assumptions 1 and 2 are reasonable approximations to make given an unknown environment, however, they are not exact. It is especially not exact for camera generated features from the same object, given that their dynamics will be strongly correlated. However, these approximations are standard practice in tracking literature and practice in order to yield a computationally tractable solution [18], [23]. Assumption 3 is justifiable if interactions between targets can be ignored. However, similarly to the first two assumptions, this approximation would not be exact with camera generated features given that the dynamics of features from the same object would be statistically correlated and therefore, not independent. In the event that targets physically interact with one another in the environment, the assumption would also not be exact. If the interactions could be ignored and the sets X_{k-1} (multi-target states at $k-1$) and Γ_k (birth set) were Poisson RFS, then Assumption 3 would be exact.

Propagating the first-order statistical moment reduces the multi-target Bayes filter to the *PHD filter* described by [47], [52]

$$\begin{aligned} v_{k|k-1}(\mathbf{x}) = &\int p_{S,k}(\boldsymbol{\varsigma}) f_{k|k-1}(\mathbf{x}|\boldsymbol{\varsigma}) v_{k-1}(\boldsymbol{\varsigma}) d\boldsymbol{\varsigma} \\ &+ \int \beta_{k|k-1}(\mathbf{x}|\boldsymbol{\varsigma}) v_{k-1}(\boldsymbol{\varsigma}) d\boldsymbol{\varsigma} + \gamma_k(\mathbf{x}) \end{aligned} \tag{2.121}$$

and

$$v_k(\mathbf{x}) = [1 - p_{D,k}(\mathbf{x})]\, v_{k|k-1}(\mathbf{x}) + \sum_{\mathbf{z} \in Z_k} \frac{p_{D,k}(\mathbf{x}) g_k(\mathbf{z}|\mathbf{x}) v_{k|k-1}(\mathbf{x})}{\int p_{D,k}(\boldsymbol{\varsigma}) g_k(\mathbf{z}|\boldsymbol{\varsigma}) v_{k|k-1}(\boldsymbol{\varsigma}) d\boldsymbol{\varsigma}}. \tag{2.122}$$

$v_{k|k-1}(\cdot)$ denotes the PHD of the state predicted RFS from $k-1$ to k, from the previous state set. $\boldsymbol{\varsigma} \in X_{k-1}$ and $\mathbf{x} \in X_k$. $v_k(\cdot)$ denotes the PHD of the measurement update RFS. $\gamma_k(\mathbf{x})$ denotes the PHD of the Poisson RFS for random target births at k and $\beta_{k|k-1}(\mathbf{x}|\boldsymbol{\varsigma})$ the PHD of the Poisson RFS for targets that spawn from previous targets. $p_{S,k}(\boldsymbol{\varsigma})$ denotes the state dependent probability of survival and $p_{D,k}(\boldsymbol{\varsigma})$ the state dependent probability of detection. $f_{k|k-1}(\cdot|\cdot)$ and $g_k(\cdot|\cdot)$ in this context operate on the single target domain, as oppose to its usage on the multi target domain in multi-target Bayes filter context. Equation 2.122 (the update stage) is composed of two terms: a missed detection term and a measurement update term. The missed detection term is used to retain PHD mass of the propagated PHD (Equation 2.121), given that certain measurements might not be generated, and is a function of the probability of missed detection $(1 - p_{D,k}(\mathbf{x}))$. The measurement update term exhaustively updates the PHD mass with every measurement in the measurement set, both target generated and clutter.

The resultant PHD filter does not suffer from the combinational complexity of the multi-target Bayes filter and operate on χ_{space}, instead of χ_{space}^{Mul}.

Equations 2.121 and 2.122 contain integrals on the single-target state space χ_{space} instead of the multi-target state space χ_{space}^{Mul} (contrasted with Equations 2.118 and 2.119), thereby avoiding the associated cross-coupled complexity of the multi-target Bayes filter and the multi-target state space χ_{space}^{Mul}. As a result, Equations 2.118 and 2.119 simplify to Equations 2.121 and 2.122 by using both the expectation of RFSs and the assumptions that assert Poisson characterised sets, and independence between target and measurement set subsets respectively.

The MHT approach is of interest for the purposes of robust tracking due to its optimal estimation, and robust handling of clutter and incorrect data association. MHT also runs the risk of being computationally taxing, but can be mitigated via PHD mass filtering techniques. MHT has also made many computational advances since its inception such as the derivation of the PHD filter from the multi-target Bayes filter. This derivation is made possible by assuming that the cardinality is Poisson distributed for RFSs, which is not made for the cardinalized-PHD (CPHD) filter [49], [72], which assumes any discretely distributed density over the cardinality of RFSs. Prior CPHD mass and cardinality distributions are interdependent in their propagation and update. The CPHD filter has computational complexity of $\mathcal{O}\left(|Z^3|\right)$ [45], due to the elementary symmetric function, despite the polynomial roots based computational advancements made to its implementation [73].

2.6.3 Overview of Data Association Techniques

An overview of GNN, PDA, JPDA, and MHT is presented in Sections 2.5 and 2.6. These techniques deal with the issue of unknown correspondence or data association, that is which targets generated which measurements. GNN makes the most likely measurement-to-target association, but runs the risk of that association being incorrect, which does not make GNN reliable. PDA results in a suboptimal estimate, but safeguards against GNN's biggest issue. However, PDA does not consider the presence of other targets, and therefore suffers from coalescence. JPDA deals with the issue of coalescence by considering other targets, but also yields a suboptimal result. JPDA suffers from combinatorial complexity as the number of targets and measurements increase. The combinatorial complexity is problematic for feature-based measurements from camera sensors, given the amount of measurements that are detected.

The derived PHD filter results in a computationally tractable implementation of the multi-target Bayes filter. This filter does not run the risk of GNN, nor suffers from coalescence or combinatorial complexity. The PHD filter deals with issues such as the unknown number of targets in the environment, missed detects, and clutter measurements. The filter has issues with implementation, since its exhaustive associations result in an unbounded number of state estimates. This issue is explored in Chapter 6.

3. Design Choices and Overview

Detection and tracking of moving objects in a robot's environment requires a few key components. Most of these have been addressed and expounded in the literature review and form the basis of the discussion in this chapter. A discussion follows that will determine which techniques should form the proposed solution, given the robustness and reliability design choices. After the proposed solution has been outlined, the following chapters will address the implementation of each of these major components. The DATMO problem is characterised and approach by two stages: first, the creation of the measurement set, and second, the utilisation of the created measurement set. Image processing techniques that can be used to create the measurement set are addressed. Filtering techniques, along with data association techniques, are addressed in order to use the measurement set, and result in robust and reliable state estimates of moving objects (targets).

3.1 Filtering and Data Association

Four data association techniques, outlined in Sections 2.5 and 2.6, were investigated. These are GNN, the PDA filter, the JPDA filter, and MHT in the form of the PHD filter. GNN, with its single hard association, would result in the least amount of computational cost. However, it is the most unreliable and non-robust given the risk of making an incorrect association. The PDA filter results in a suboptimal result, but does not run the risk of making a single incorrect association. As a result, the PDA filter is more robust than GNN. The JPDA filter would be more robust than the GNN technique and solves the coalescence issue related to the PDA filter, but is still suboptimal in estimation accuracy, given the weighted pseudo-measurement update. Despite the suboptimal estimation result, the JPDA filter is a desirable technique given its congruence and elegance as a multi-target variant of the PDA filter. However, the computational burden of accumulatively summing the conditional probabilities over all data association possibilities suffers from combinatorial complexity. As the number of targets and the number of measurements increase, the possible combinational outcomes increase exponentially [36], as expressed by Equation 2.110. As a result, the JPDA filter becomes too expensive to implement.

Implementation Choice

The data association technique chosen is the PHD filter as part of the MHT framework. The PHD filter resolves the data association problem, in the presence of clutter measurements, by making all possible measurement-to-target associations. The filter deals with the issue of an unknown number of targets in a robot's environment by modelling uncertainty with RFSs. A RFS does not regard the order in which elements appear in the set, nor the amount of elements in the set. The filter is robust to clutter and will reliably make the desired measurement-to-target associations. The filter also addresses the issue of missed detections by modelling a missed detection term, as a function of the probability of detection, making the filter robust and reliable for DATMO.

The PHD filter is implemented using a density-based description of the intensities of the RFSs. The chosen densities are Gaussian mixtures (GMs). The derivation and expression of the GM density-based description of the PHD filter is done in Chapter 5. The chosen framework is optimal in estimation, but could be more computationally intensive than desired. The issue of an unbounded number of estimates has to be mitigated, or at least addressed in order to making the implementation possible and reduce the computational burden. The unbounded estimates issue is addressed in Chapter 6.

3.2 Image Processing

An alternative to the feature-based approach, expounded in literature, is a *disparity map* [60] approach. Image subregions of different sizes from two stereo rectified images are correlated along horizontal epipolar lines in order to determine their corresponding subregions. The amount of flow, the shifted distance in the correlation process, is the disparity and indicates the depth of the object (or image subregion). Figure 3.1 displays the disparity between two stereo rectified images, with darker areas correlating to smaller disparity and therefore longer distances. Figure 3.2 is the same image, but filtered with smoothing and linear interpolation, especially around object borders. This approach was not used for DATMO due to the computational cost of the method. Extracting usable information from the resultant image would also add to already undesirable computational burden.

A similar flow based approach called *optical flow* [41], could also be used in order to use vision-based sensors. This method is similar, but instead of correlating between stereo rectified images, it correlates between consecutive images from the same camera. This method determines how the image flows between these consecutive scenes. Using stereo vision, this approach can be extended to *scene flow* [69], which describes how a point flows in the state space. This method was also not used due to the computational cost. Optical flow is more expensive than disparity flow, since disparity flow correlates image subregions across one

dimension, that is horizontally if stereo rectified, while optical flow correlates across two dimensions, that is both image plane dimensions. It is also difficult to account and correct for the ego motion of the robot, that is the robot's own motion, since ego motion causes relative scene flow. Given the computational cost and difficulties involving ego motion, a feature-based approach is used instead.

Figure 3.1: Unfiltered disparity flow of stereo rectified images.

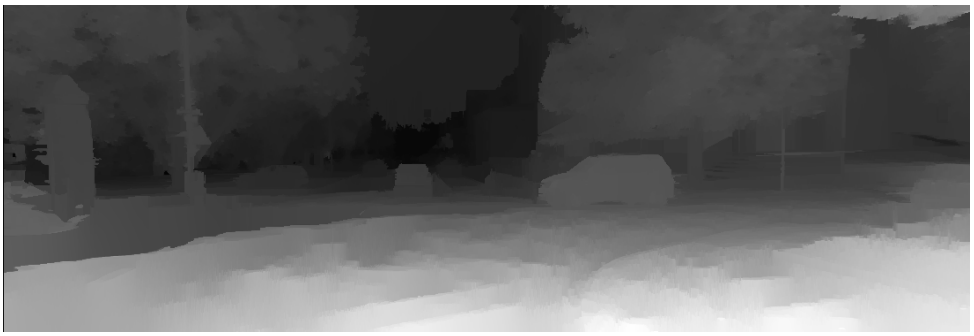


Figure 3.2: Filtered disparity flow of stereo rectified images.

Implementation Choice

The most popular feature detection algorithms are expound in the literature review. These include FAST, ORB, KAZE, A-KAZE, SIFT, and SURF. FAST features are not invariant to scale and rotation, resulting in an unreliable methodology. The SIFT and SURF algorithms are patent protected and the authors require license fees for using their algorithms. Given these limitations, the attention shifts to the KAZE, A-KAZE, and also ORB. The licensing on SIFT is not an issue since KAZE is expected to outperform SIFT due to its nonlinear scale space method. ORB, KAZE, and A-KAZE will be inspected for their robustness and reliability. A-KAZE will undoubtedly be faster than KAZE, but its performs relative to ORB would be of interest. ORB's accuracy in comparison with KAZE and A-KAZE would be of interest given its lack of subpixel localisation. KAZE uses float descriptors, that are a function of gradients, while ORB and A-KAZE use binary descriptors, that are a function of pixel

intensity binary test results. The relative impact of float descriptors and binary descriptors on performance would be of interest in order to determine the robustness of the different description techniques. Both KAZE and A-KAZE use nonlinear diffusion techniques to create the scale space, while ORB uses a linear Gaussian approach. KAZE, A-KAZE, and ORB will be implemented and comparatively analysed.

KAZE and A-KAZE are compared to one another, since both are expected to out perform SIFT [13], which has become something of a benchmark technique, but both use different descriptors. KAZE uses a local gradient-based descriptor which is represented by a float vector, whereas A-KAZE uses the results of binary test outcomes that are a function of intensities and is represented by a binary sequence. It is expected that A-KAZE will be computationally faster given the computational advances that define it, but the performance of these feature detectors are often dependent on the application for which they are used.

ORB is used for its unique approach to detecting features, which is not gradient based with subpixel accuracy like most approaches, but rather intensity-based without subpixel accuracy. ORB features could be augmented for subpixel accuracy via curve fitting and interpolation, but was left unaltered in order to test the base algorithm with those that do have subpixel accuracy. FAST, which forms part of the foundation of ORB as oFAST, has been shown to function better in environments characterised by fewer gradients [38], such as an image of a planar environment with low texture variances, due to its intensity based testing. ORB was also chosen as a feature detection method in order to compare its computational performance with A-KAZE. Comparing ORB and A-KAZE on the basis of computational speed in a filtering context would be insightful, given that both ORB and A-KAZE are known for their speed. The implementation of ORB, KAZE, and A-KAZE are outlined in chapter 4, which addresses issues such as feature matching.

3.3 Overview

Figure 3.3 (on its own page) depicts a simplified overview of the design. Features are detected, matched, and vetted for reliability and the resultant feature matches form the measurement set. Next, a density based PHD filter is used in order to obtain state estimate distributions. And finally, the resultant distributions are transposed into an inertial coordinate axis to be used in the prediction of states in the PHD filter at the next time instant.

A transformation model is derived that describes the relationship between the visual measurements and the 3D positions of objects in the environment via triangulation. Stereo vision is used to obtain depth estimates, with triangulation, without first creating a virtual baseline with movement. This transform is derived using the pinhole camera model. This model is

described with parameters that are determined via camera calibration. Stereo images are rectified to account for the relative pose of cameras. Measurements are used as the representation of the environment via feature detection and matching, outlined in Chapter 4, by using KAZE, A-KAZE, and ORB. Feature matches are used to form the measurement set in the Bayesian filtering framework. The data association problem between object states and measurements can be addressed by a variety of means, of which MHT in the form of the PHD filter was determined to be the most favourable. The PHD filter is implemented with a density-based description, outlined in Chapter 5, and the resultant unbounded number of estimates issue, and the techniques used to address the issue, are outlined in Chapter 6. Because of the ego motion of the robot, at each time step the state estimates and measurements are described with respect to different coordinate systems. These need to be transposed into the same coordinate description in order to update the state estimates.

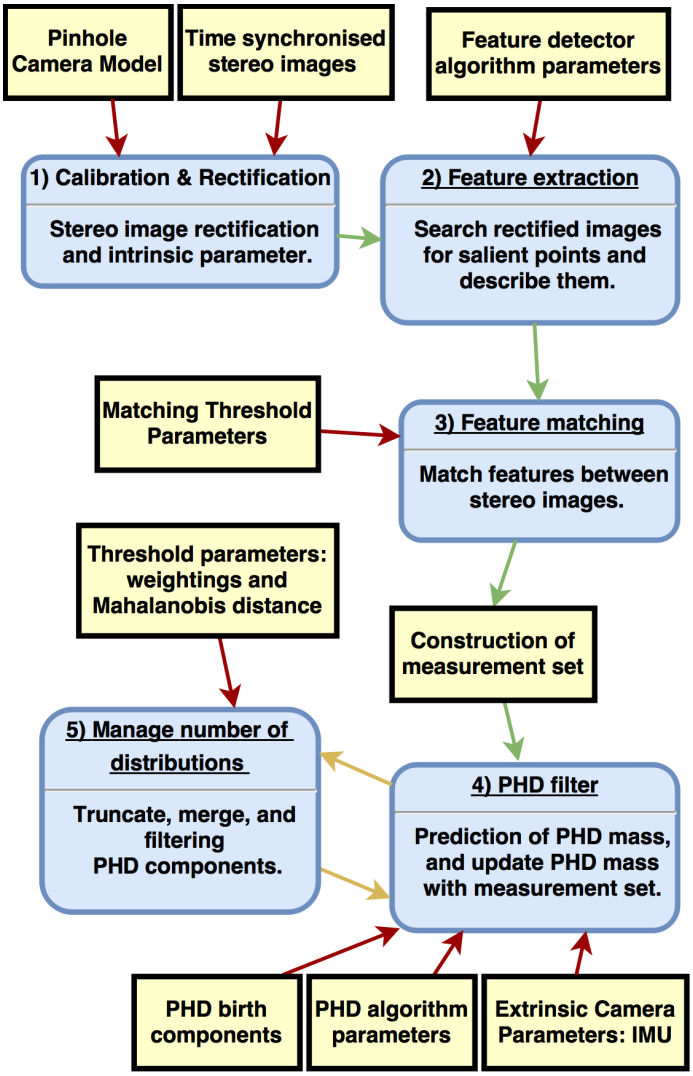


Figure 3.3: Overview of the sequence of events that take place for DATMO functionality. Numbered blue blocks indicate the flow of the process, and yellow inputs to the process.

4. FEATURE HANDLING

This chapter addresses the methods used to create the measurement set. These methods include how the horizontal epipolar and positive disparity constraints are used to match features. The literature review section outlines the most popular feature detection algorithms and some of the earlier methods that paved the way for them. Of these approaches, ORB, KAZE, and A-KAZE were selected for testing in a state estimation filtering framework. Measurement error is also addressed in this chapter. The following section will focus on how the feature detectors are used and differ.

4.1 Detecting Features

Figures 4.1, 4.2, and 4.3 depict detected features in the same image from the KITTI dataset. The individual colours have no significance other than being distinct and clear. The features in Figures 4.1, 4.2, and 4.3 are obtained using ORB, KAZE, and A-KAZE feature detection respectively. These figures demonstrate how gradient based detectors (KAZE and A-KAZE) don't function well in regions with a lack of clear gradients. KAZE and A-KAZE do not yield many feature responses from the black vehicle on the right, while ORB yields more responses in this region. ORB does not yield many responses in the region with the white vehicle on the left, while KAZE and A-KAZE yield quite a few more features in the same region.

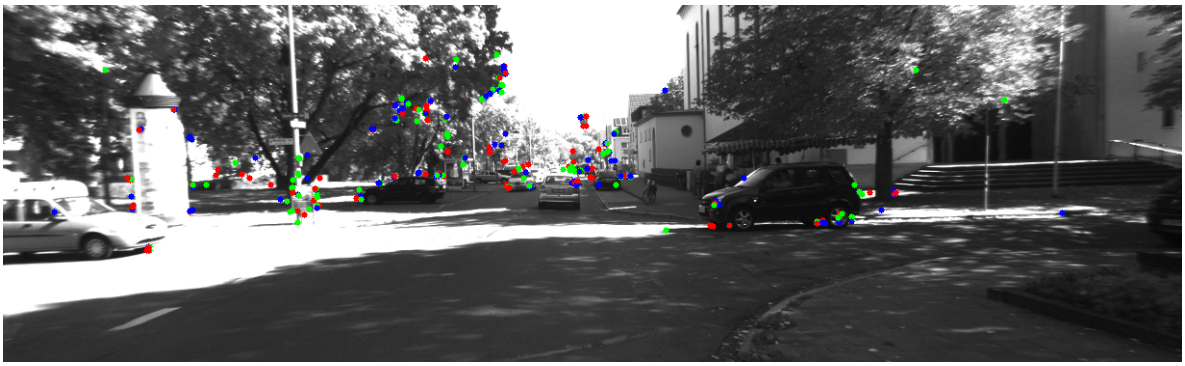


Figure 4.1: ORB feature detection from KITTI dataset image.

The number of detected features in an image can be controlled by adjusting the parameters of

Figure 4.2: KAZE feature detection from KITTI dataset image.

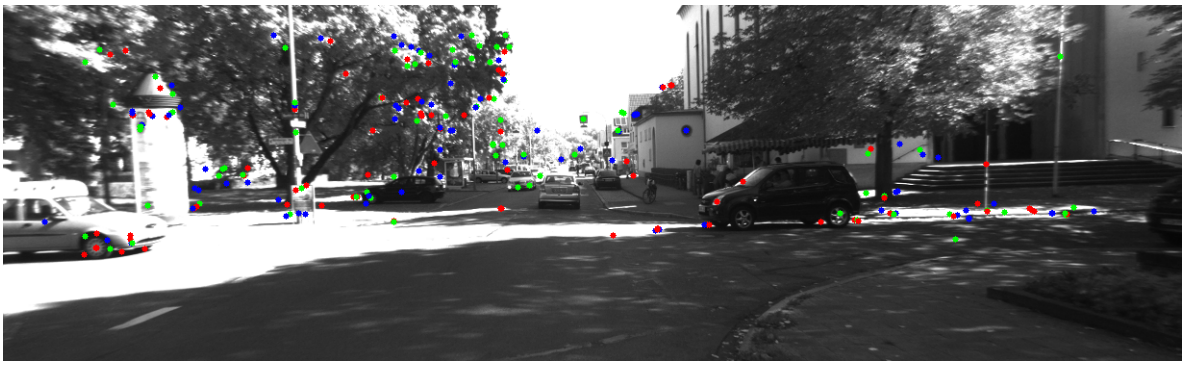


Figure 4.3: A-KAZE feature detection from KITTI dataset image.

the each feature detection algorithms, or by simply discarding some the features if a maximum limitation is desired. The size of the scale space influences the number of feature as each scale level is searched. Increased scale space resolution results in more features, but increases the computational burden. The number of ORB features is influenced by the intensity threshold of the FAST algorithm. Lower testing thresholds increases the number of features, but increases the risk of introducing weak, non-robust features that lack distinction. An increased number of features ensures that the surveillance region of the camera's field of view is thoroughly inspected and represented in the measurement set, but increases the computation which is a cumbersome problem in the MHT framework. However, a lack of features increases the risk that some objects in the environment might not be detected.

Gradient based methods often yield multiple responses in undesirable high gradient regions such as tree branches as seen in Figures 4.2 and 4.3. These features (KAZE and A-KAZE) do not contribute to DATMO functionality and increase the computational burden. Many of these will be wasted on undesirable gradient responses, which in turn results in less coverage of desirable regions containing moving objects, if a fixed number of features are detected. We attempted to address this problem by using a grid-based approach, depicted in Figure 4.4. Using this method, each grid subregion is searched in isolation from the rest of image to ensure a more equal coverage of the surveillance region, thereby restricting some of the undesirable

feature responses.

However, attempting to detect 300 features using the grid-based approach would require 4 features to be detected for each of the 70 subregions depicted in Figure 4.4. This approach would result in only 4 features obtained from the red marked subregion, which contains two moving objects. These 4 features would result in poor representation of the vehicles and therefore more feature would need to be obtained from each subregion. Ultimately, this approach was abandoned given the sheer number of features that would need to be extracted from each subregion. Instead, the features from a global detection would need to be sufficiently distinct in comparison with the undesirable features. These undesirable features tend not to be very repeatable, that a specific points do not consistently appear in a sequence of images given their sensitivity to changes in view point. A grid-based approach could be revised with the intention of choosing an optimal amount of subregions.

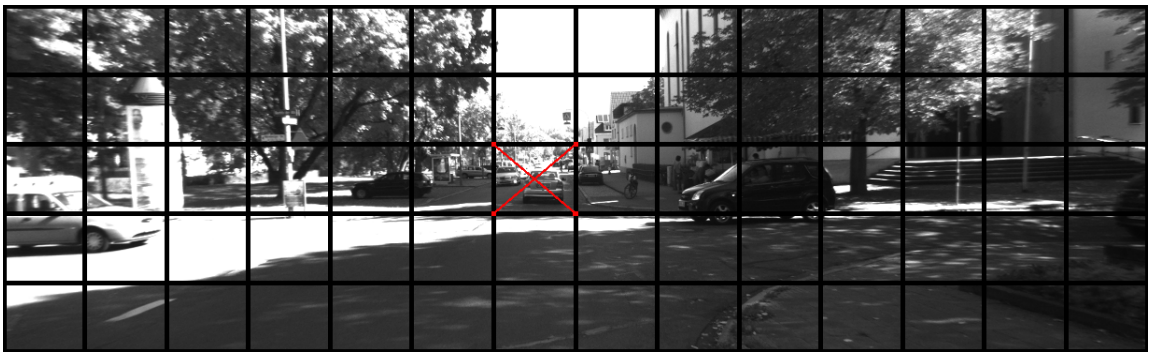


Figure 4.4: Grid based approach to feature detection. The image is divided into 70 subregions which are searched independently. The red marked subregion contains 2 moving objects.

4.2 Matching Features

Once features are extracted from the left and right camera images, they are matched in order to form matched pairs that can be triangulated, and used for state estimation. Feature matching is done by testing and comparing a distance metric that is a function of the descriptors of features. The smallest distance pair is chosen as a match. This distance metric indicates the similarity of the matched features, and hence the matching strength and reliability. Smaller distances indicate more similarity and are therefore more reliable matches. Distances that exceed a specified threshold are discarded which reduces the number of unreliable matched pairs, and also results in a computational gain. The matching strategy that we use matches features both ways between images and only selects pairs that match both ways. A feature α needs be matched to β both when starting with α and searching for a match, and when starting with β and searching for a match. This increases the reliability and robustness of matches and is an alternative to the ratio test used by Lowe [44].

Float vector descriptors use the *Euclidean distance* metric, while binary sequences use the *Hamming distance* metric. The Euclidean distance is calculated using the sum of square differences of the descriptors. The Hamming distance, between sequences of the same size, is the number of differences between their symbols, typically binary in probabilistic robotics. In information theory, this result is the number of minimum changes that would need to be made to match the two sequences exactly [39]. The Hamming distance between binary sequences "1001" and "1010" would be 2, due to the differences in the last two symbols. A maximum of 300 feature matches have been chosen in order to reduce the amount of computation. Feature matches are ranked by their distance metric, and the smallest 300 pairs are chosen, assuming that more than 300 were in fact obtained.

Two constraints are also used in the matching process, namely the epipolar constraint and the positive (non-negative) disparity constraint. The epipolar constraint is the result of stereo rectification, which constrains a feature's match along the horizontal epipolar line which is at the same vertical height in the corresponding image plane. The positive disparity constraint dictates that the disparity of the match $x_{pL} - x_{pR}$ has to be positive. The depth of the detected object from the cameras into the environment, perpendicular to the baseline, is inverse proportional to the disparity $x_{pL} - x_{pL} \propto \frac{1}{z}, z \in Z_c$. Given that the object is in the field of view of the cameras, and therefore in front of the cameras, z has to be positive and therefore $x_{pL} - x_{pR}$ has to be positive. These constraints limit the number of viable feature matches, resulting in computational advances since only features that meet the constraint criteria need to be inspected. Figure 4.5 shows an example of stereo matched features from two zoomed-in left-right images from the KITTI dataset.

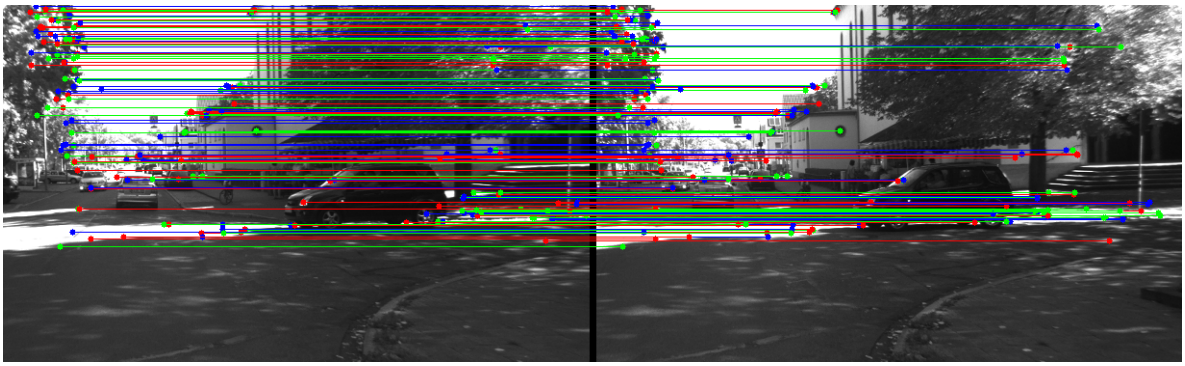


Figure 4.5: Example of stereo feature matching using epipolar and positive disparity constraints. The lines indicate the feature matches between the left and right images.

4.3 Measurement Set Error

The matched features form the set of measurements that are used to estimate the states of moving objects. However, these measurements are influenced by different sources of error.

These errors reduce the accuracy of the feature measurements which negatively influences the 3D triangulation, and hence influences the accuracy of the resultant state estimates. Two sources of error are feature accuracy error and feature matching error.

Feature Mismatch Error

The measurement set can be expressed as the union of all inlier measurements, correctly matched features as a set ${Z_k}^{in}$, and outlier measurements, mismatched features as a set ${Z_k}^{out}$. This can be expressed as

$$Z_k = {Z_k}^{in} \bigcup {Z_k}^{out}. \tag{4.1}$$

The constrains contained in this chapter function to reduce the amount of outliers, that is feature mismatches. The result is that the mismatched set cardinality $|{Z_k}^{out}|$ is reduced. However, even given all of the methods outlined in this chapter to ensure robust and reliable feature matches, mismatches (matching errors) will still occur occasionally. The PHD filter, being a MHT technique, should be able to handle any potential mismatches that make its way into the measurement set.

Feature Accuracy Error

The other source of error is the accuracy of feature locations, which influences the inlier set ${Z_k}^{in}$, as well as the outlier set. The accuracy is influenced by both camera sensor noise, and the accuracy of the feature detection method. The accuracy with which features are located within their respective left and right images before matching will influence the 3D triangulated potion in the state space. The triangulation equations, from the literature review, with added pixel noise can be expressed as

$$x = \frac{B(x_{pL}^{noise} - c_x)}{x_{pL}^{noise} - x_{pR}^{noise}},\ x \in X_c \tag{4.2}$$

$$y = \frac{B(y_{p}^{noise} - c_y)}{x_{pL}^{noise} - x_{pR}^{noise}},\ y \in Y_c \tag{4.3}$$

and

$$z = \frac{fB}{x_{pL}^{noise} - x_{pR}^{noise}},\ z \in Z_c. \tag{4.4}$$

These equations express how a matched feature pair $[x_{pL}^{noise},\ x_{pR}^{noise},\ y_{p}^{noise}]^T$, in the measurement space, transforms (triangulates) into the state space as $[x,\ y,\ z]^T$. x_{pL}^{noise} and x_{pR}^{noise}

denote the left and right horizontal pixel locations with noise, that is $x_{pL}^{noise} = x_{pL} + n_{xpL}$ and $x_{pR}^{noise} = x_{pR} + n_{xpR}$, where n_{xpL} and n_{xpR} denote noise added to the true values. y_p^{noise} denotes the matched vertical pixel position with noise. y_p^{noise} is a function of the left and right vertical feature locations, that is y_{pL} and y_{pR}. Each of these vertical pixel positions, coming from different cameras, contribute their own noise, n_{ypL} and n_{ypR}. As a result, the matched vertical pixel position can be expressed as the average between the two positions, that is

$$\begin{aligned} y_p^{noise} &= \frac{1}{2}(y_{pL}^{noise} + y_{pR}^{noise}) = \frac{1}{2}(y_{pL} + n_{ypL} + y_{pR} + n_{ypR}) \\ &= \frac{1}{2}(y_{pL} + y_{pR}) + \frac{1}{2}(n_{ypL} + n_{ypR}) \qquad (4.5) \\ y_p^{noise} &= y_p + \frac{1}{2}(n_{ypL} + n_{ypR}) = y_p + n_{yp}, \qquad (4.6) \end{aligned}$$

with $y_p = \frac{1}{2}(y_{pL} + y_{pR})$ and $n_{yp} = \frac{1}{2}(n_{ypL} + n_{ypR})$. These noise terms, n_{xpL}, n_{xpR}, and n_{yp}, model deviations from the true value. They are influenced by sensor noise and feature location accuracy. The influence of noise on the resultant 3D triangulated position is of interest in order to determine the sensitivity of measurements to noise. Figure 4.6 demonstrates how linear incremental changes of the points whether the projection lines intersect the image planes, result in nonlinear triangulated 3D positions.

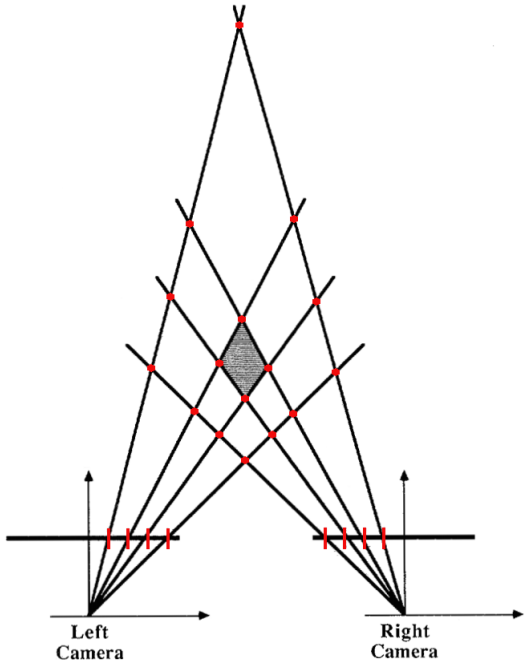


Figure 4.6: Illustration of nonlinear triangulation disparity. The further away an object is, the larger the expected triangulation error. Image reproduced with permission from Matthies [50].

Figure 4.6 also shows how distant objects are more sensitive to noise than closer ones, and

therefore to triangulation error. Small derivations, from the true projected pixel positions of distant objects, cause much larger triangulation error than the same small derivations would cause for closer objects.

Figure 4.7 demonstrates the result of transforming a matched feature pair with different noise samples drawn from a Gaussian distribution. The resultant point cloud demonstrates the expected distribution when a Gaussian is subject to the nonlinear triangulation transform. Samples are drawn from a Gaussian $\mathcal{N}(\mathbf{0},\ \Sigma_{noise})$, with $\Sigma_{noise} = \text{diag}[\sigma_x^2,\ \sigma_x^2,\ \sigma_y^2]$. The assumption is that the variances of the noise influencing the pixel positions $[x_{pL},\ x_{pR},\ y_{pL},\ y_{pR}]^T$ are equal, with a value of σ^2. Therefore $\sigma_x^2 = \sigma^2$, and $\sigma_y^2 = 0.5\sigma^2$, given that the matched vertical position y_p is the average of two independent vertical positions y_{pL} and y_{pR}.

The cameras in Figure 4.7 are located on the X_C axis looking toward the positive direction of the Z_C axis, with the middle point of the baseline at $X_C = 0$. The relative axis divisions and the degree to which the image is zoomed in are both important and needs to be noted. The figure depicts what would be characteristic of feature accuracy error. If a feature is not accurately localised in its image, then the triangulation process results in a potential range of locations, as depicted by Figure 4.7. ORB, not using subpixel accurate techniques, is expected to result in larger error after the triangulation process. The camera projection transform is characterised by large uncertainty in the dimension of the projection line, and much smaller uncertainty elsewhere. The projection line in Figure 4.7 is horizontal, hence the resultant large uncertainty (point spread) along the Z_C axis, and the relative certainty along the X_C axis.

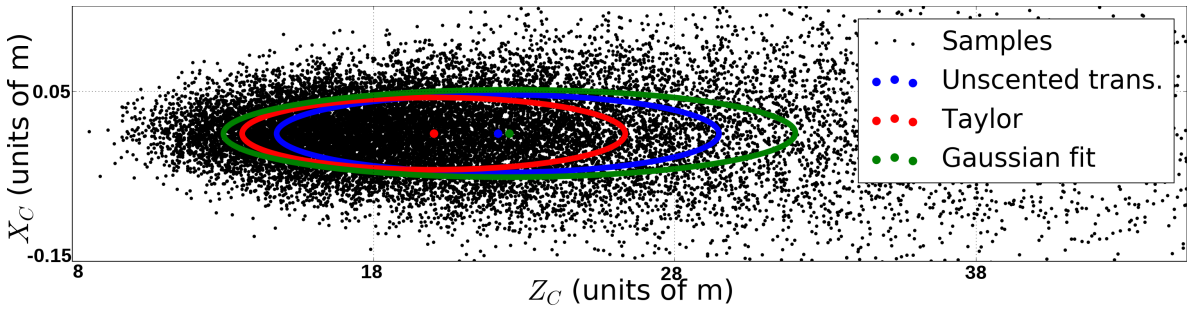


Figure 4.7: Triangulation transform of Monte Carlo feature match pair samples from a Gaussian distribution in measurement space, into state space, displayed as 2 dimensions. Note the axis division step sizes. The red ellipse denotes a first-order Taylor approximation, the blue ellipse a unscented transform approximation, and the green ellipse the actual spread of the samples.

The resultant distribution in Figure 4.7, as a result of the nonlinear transform, is not a distribution that can be described with an analytical expression characterised by parameters. The chosen Bayesian framework assumes Gaussian distributed measurement noise, which transforms to the depicted distribution, therefore the resultant distribution needs to be approximated

with a nonlinear approximation technique. The blue and red ellipses in Figure 4.7 denote confidence bounds of Gaussian approximations of the resultant distribution. The red denotes Taylor linearisation while the blue denotes the unscented transform. The green represents the spread of the actual sample points. The figure shows that the unscented transform approximation is more representative of the transform than the Taylor linearisation, given that it is more similar to the green ellipse than the Taylor approach, and is therefore the chosen method.

The accuracy of feature detection algorithms, especially ORB, is still an issue. An approximation of the expected deviation error z^{σ}_{error} in the triangulation process, due to feature accuracy error, is proportional to the square of the actual distance of the object [35], as expressed by

$$z^{\sigma}_{error} \propto \frac{z^2 \alpha_{CCD}}{Bf}, \tag{4.7}$$

where α_{CCD} is the size of a camera's CCD sensor and z is the true distance. As the baseline and focal length increase, the error decreases, and as the true distance of the point increases, the error increases quadratically. As a result, the further away an object is, the larger the expected triangulation error as depicted by Figure 4.6.

In conclusion, the unscented transform will be used as an approximation of the nonlinear triangulation transform. Mismatched features that form part of the measurement set will be managed with the PHD framework and its characteristic dynamics that can handle clutter measurements.

5. GAUSSIAN-MIXTURE PHD FILTER

Once features are obtained using one of many features detectors, they form the measurement set that is used to obtain state estimates. This set is created by detecting and matching features from left and right stereo rectified images. In Chapter 4 many of the constraints that reduce the amount of mismatches are outlined. Chapter 4 also highlights the issue of measurement noise and how it is influenced by the nonlinear triangulation transform. The implementation of the PHD filter is expounded in this chapter, which involves a density-based expression of the intensity quantities.

Once the expression is in place, the measurement set is created using matched feature pairs which can be used in the filter context to result in state estimates. These state estimates are of moving 3D points in the environment, and give an indication of the motion of the moving objects in the environment. The density-based expression in this section is reproduced from Vo and Ma's 2006 paper [52].

Originally, Monte Carlo statistical sampling approaches were used to implement the PHD filter [47], which approximates the integrals with samples from the distributions [16]. This approach has several drawbacks such as computational issues. There is also an issue of accuracy. Samples are not necessarily located at a local maxima of the PHD mass [52]. These issues were addressed by modelling the PHD mass with a Gaussian mixture (GM) density-based description, resulting in the *Gaussian-mixture PHD filter* (GM-PHD) by authors Vo and Ma in 2006 [52].

The intensities (PHDs) are modelled with sums of weighted Gaussian components that approximately represent the PHD masses, or at least the dominant mass volumes. The weights represent the results of integrating over the relevant PHD regional Gaussian components, which in turn represents the expected number of elements in the RFSs in the region over which is being integrated, as described by Equation 2.120. The original expression also made the assumption that the likelihood function and transition density are linear transforms, but can be addressed with a nonlinear approximation technique. The density-based description alleviates some of the computational burden and the Gaussian components yield very definite local maxima in the PHD mass at the Gaussian means. The resulting filter functions similarly to the well-known prediction-update framework as expressed in the KF framework.

5.1 Assumptions

In order to arrive at the desired density-based description, some assumptions are made in the PHD context as derived in Chapter 2. The transition density and likelihood functions are assumed to be linear transforms with additive, zero-mean, independent Gaussian noise. The transition density is described by

$$f_{k|k-1}(\mathbf{x}|\boldsymbol{\varsigma}) \sim \mathcal{N}(\mathbf{x};\, \mathbf{F}_{k-1}\boldsymbol{\varsigma},\, \mathbf{Q}_{k-1}). \tag{5.1}$$

The transition density expresses the motion model in the Bayesian framework, and describes how states evolve from one moment to the next. The likelihood function is described by

$$g_k(\mathbf{z}|\mathbf{x}) \sim \mathcal{N}(\mathbf{z};\, \mathbf{H}_k\mathbf{x},\, \mathbf{R}_k), \tag{5.2}$$

and characterises the transform between the measurements and the 3D world. These descriptions use similar notation as expressed in the KF section of the literature review. The equations outlined in this section reflect the linear assumption since Vo and Ma's standard description assumes linear transforms. However, the implementation of the GM-PHD framework was done using the unscented transform, as outlined in the literature review, due to the nonlinear triangulation transform. Chapter 4 demonstrates that the unscented transform results in a more accurate Gaussian approximation of the nonlinear triangulation transform than the first-oder Taylor approximation.

The probability of survival $p_{S,k}(\mathbf{x})$ (Equation 2.121) and probability of detection $p_{D,k}(\mathbf{x})$ (Equation 2.122) are assumed to be state independent, as described by

$$p_{S,k}(\mathbf{x}) = p_{S,k} \tag{5.3}$$

and

$$p_{D,k}(\mathbf{x}) = p_{D,k}. \tag{5.4}$$

It would be advantageous to make $p_{S,k}(\mathbf{x})$ and $p_{D,k}(\mathbf{x})$ state dependent in a way that is representative of the application. $p_{D,k}(\mathbf{x})$ would be influenced by the strength of features and matches, the more reliable and repeatable feature matches are, the higher the probability of detection. Especially for highly repeatable feature matches. $p_{D,k}(\mathbf{x})$ is also influenced by the stereo camera application. $p_{D,k}(\mathbf{x})$ could be expressed as a function of an object's position and direction of movement in the field of view of the cameras. If a moving object is near the edge of the field of view or predicated to be outside the field of view, then $p_{D,k}(\mathbf{x})$ would be smaller than a situation where an object is not moving in the middle of the field of view. $p_{S,k}(\mathbf{x})$ would usually be large, but occlusion would result in an object disappearing, or 'not surviving', using

PHD filter jargon. The frequency of occlusion is difficult to know or to predict, but if the extent of objects are known, then some occlusion could theoretically be predicted. Extent would need to be determined, via clustering methods, to allow such an objective. These concepts are beyond the scope of the design approach, so $p_{S,k}(\mathbf{x})$ is assumed to be state independent.

5.2 Filter Equations

The predication-update equations of the GM-PHD filter follows from these assumptions and expressions. The posterior GM-PHD at $k-1$ is represented by

$$v_{k-1}(\mathbf{x}) = \sum_{i=1}^{J_{k-1}} w_{k-1}^{(i)} \mathcal{N}(\mathbf{x};\, \mathbf{m}_{k-1}^{(i)},\, \Sigma_{k-1}^{(i)}), \tag{5.5}$$

where J_{k-1} represents the number of Gaussian components that constitutes the posterior intensity. $w_{k-1}^{(i)}$ represent the relevant Gaussian weighting factors. The GM representations of the birth and spawn PHD mass components are expressed by

$$\gamma_k(\mathbf{x}) = \sum_{i=1}^{J_{\gamma,k}} w_{\gamma,k}^{(i)} \mathcal{N}(\mathbf{x};\, \mathbf{m}_{\gamma,k}^{(i)},\, \Sigma_{\gamma,k}^{(i)}) \tag{5.6}$$

and

$$\beta_{k|k-1}(\mathbf{x}|\boldsymbol{\varsigma}) = \sum_{j=1}^{J_{\beta,k}} w_{\beta,k}^{(j)} \mathcal{N}(\mathbf{x};\, \mathbf{F}_{\beta,k-1}^{(j)}\boldsymbol{\varsigma} + \mathbf{d}_{\beta,k-1}^{(j)},\, \Sigma_{\beta,k-1}^{(j)}) \tag{5.7}$$

respectively, where $J_{\gamma,k}$ and $J_{\beta,k}$ denotes the number of Gaussian components that constitutes the birth and spawn PHDs. The spawned PHD $\beta_{k|k-1}(\mathbf{x}|\boldsymbol{\varsigma})$, obtained from the $B_{k|k-1}(\cdot)$ RFS in Equation 2.116, represents new Gaussian components that are created from existing Gaussian components, but in a way that deviates from the expected state transition using the $\mathbf{d}_{\beta,k-1}$ term. The $\mathbf{d}_{\beta,k-1}$ term offsets the predicated states. $\mathbf{F}_{\beta,k-1}$ and $\mathbf{d}_{\beta,k-1}$ denote the motion model that characterises this specific state transition. $w_{\beta,k}^{(j)}$ denotes the weighting associated with the j-th Gaussian component of the spawn PHD. The spawn PHD will be neglected for the purpose of the proposed thesis application since it does not model expected behaviour in the environment.

The birth PHD $\gamma_k(\cdot)$, obtained from the Γ_k RFS in Equation 2.116, represents new Gaussian components that are randomly introduced, that is not a function of any previous Gaussian component and therefore unexpected or unpredictable. $w_{\gamma,k}^{(i)}$ denotes the weighting associated with the j-th Gaussian component of the birth PHD. $\gamma_k(\cdot)$ will be used to initialise new Gaussian components in order to initialise the state estimation process.

Propagation Equations

The propagation of all previous Gaussian components, from the PHD prediction expressed in Equation 2.121, are represented by

$$v_{k|k-1}(\mathbf{x}) = v_{S,k|k-1}(\mathbf{x}) + v_{\beta,k|k-1}(\mathbf{x}) + \gamma_k(\mathbf{x}). \tag{5.8}$$

$v_{\beta,k|k-1}(\cdot)$ is the spawn PHD, which was not used but is expressed for the purpose of completion. The spawn PHD, using Equations 5.1, 5.5, and 5.7, is expressed by

$$v_{\beta,k|k-1}(\mathbf{x}) = \sum_{j=1}^{J_{k-1}} \sum_{l=1}^{J_{\beta,k}} w_{k-1}^{(j)} w_{\beta,k}^{(l)} \mathcal{N}(\mathbf{x};\, \mathbf{m}_{\beta,k|k-1}^{(j,l)},\, \Sigma_{\beta,k|k-1}^{(j,l)}), \tag{5.9}$$

where the mean is predicated by

$$\mathbf{m}_{\beta,k|k-1}^{(j,l)} = \mathbf{F}_{\beta,k-1}^{(l)} \mathbf{m}_{k-1}^{(j)} + \mathbf{d}_{\beta,k-1}^{(l)}, \tag{5.10}$$

and the covariance by

$$\Sigma_{\beta,k|k-1}^{(j,l)} = \mathbf{Q}_{\beta,k-1}^{(l)} + \mathbf{F}_{\beta,k-1}^{(l)} \Sigma_{\beta,k-1}^{(j)} (\mathbf{F}_{\beta,k-1}^{(l)})^T. \tag{5.11}$$

The *survival* PHD $v_{S,k|k-1}(\cdot)$ is a function of the probability of survival. The propagation equations of the surviving Gaussian components are expressed by

$$v_{S,k|k-1}(\mathbf{x}) = p_{S,k} \sum_{j=1}^{J_{k-1}} w_{k-1}^{(j)} \mathcal{N}(\mathbf{x};\, \mathbf{m}_{S,k|k-1}^{(j)},\, \Sigma_{S,k|k-1}^{(j)}), \tag{5.12}$$

where the mean is predicated by

$$\mathbf{m}_{S,k|k-1}^{(j)} = \mathbf{F}_{k-1} \mathbf{m}_{k-1}^{(j)}, \tag{5.13}$$

and the covariance by

$$\Sigma_{S,k|k-1}^{(j)} = \mathbf{Q}_{k-1} + \mathbf{F}_{k-1} \Sigma_{k-1}^{(j)} \mathbf{F}_{k-1}^T. \tag{5.14}$$

The survival PHD represents all of the predicted state estimates from the previous posterior. The resultant weighting factors, after the state prediction, are $p_{S,k} w_{k-1}^{(j)}$, which represents a reduction of confidence in the region over which the Gaussian is distributed as a function of the probability of survival. The weighting factor, representing the expected number of tracked targets, reduces since some of those targets did not 'survive', as model by the probability of survival.

Equation 5.12 is the propagation of the previous posterior PHD Gaussian components, where each Gaussian component's propagation is done with equations 5.13 and 5.14, which are the same equations used by the standard linear KF. The similarities between the GM-PHD filter and the KF, and the multi-target Bayes filter and the Bayes filter, are not forced or introduced, but is the result of derivation. The simplification from Bayes filter to KF is analogous to the simplification from multi-target Bayes filter to GM-PHD filter.

Update Equations

After all of the Gaussian components are propagated they need to be updated with the measurement set Z_k. All the resultant propagated Gaussian components after the prediction stage described by Equation 5.8, with $J_{k|k-1}$ indicating the number of Gaussian components, is expressed by

$$v_{k|k-1}(\mathbf{x}) = \sum_{i=1}^{J_{k|k-1}} w_{k|k-1}^{(i)} \mathcal{N}(\mathbf{x};\, \mathbf{m}_{k|k-1}^{(i)},\, \Sigma_{k|k-1}^{(i)}), \tag{5.15}$$

where $w_{k|k-1} = p_{S,k} w_{k-1}$. The measurement update of the propagated Gaussian components results in a GM-PHD, from the PHD update Equation 2.122, expressed by

$$v_k(\mathbf{x}) = (1 - p_{D,k}) v_{k|k-1}(\mathbf{x}) + \sum_{\mathbf{z} \in Z_k} v_{D,k}(\mathbf{x}; \mathbf{z}), \tag{5.16}$$

given the measurement set Z_k. Equation 5.16 represents the posterior GM-PHD after the update stage which comprises two terms. The first is the missed detection term, which is a function of the probability of a missed detection $(1 - p_{D,k})$. This term allows for propagated existing components that are not detected to not simply be discarded, but to be retained with its weighting factors reduced. The missed detection term highlights the robustness of the PHD filter framework. The second term is the update term using the measurement set which is created via feature detection. This GM-PHD is expressed as

$$v_{D,k}(\mathbf{x};\, \mathbf{z}) = \sum_{j=1}^{J_{k|k-1}} w_k^{(j)}(\mathbf{z}) \mathcal{N}(\mathbf{x};\, \mathbf{m}_{k|k}^{(j)}(\mathbf{z}),\, \Sigma_{k|k}^{(j)}). \tag{5.17}$$

The means and covariances are updated by

$$\mathbf{m}_{k|k}^{(j)}(\mathbf{z}) = \mathbf{m}_{k|k-1}^{(j)} + \mathbf{L}_k^{(j)} (\mathbf{z} - \mathbf{H}_k \mathbf{m}_{k|k-1}^{(j)}) \tag{5.18}$$

and

$$\Sigma_{k|k}^{(j)} = [I - \mathbf{L}_k^{(j)} \mathbf{H}_k]\, \Sigma_{k|k-1}^{(j)}, \tag{5.19}$$

using the Kalman gains

$$\mathbf{L}_k^{(j)} = \Sigma_{k|k-1}^{(j)} \mathbf{H}_k^T \left(\mathbf{H}_k \Sigma_{k|k-1}^{(j)} \mathbf{H}_k^T + \mathbf{R}_k \right)^{-1}. \tag{5.20}$$

The updated weighting factors are calculated as

$$w_k^{(j)}(\mathbf{z}) = \frac{p_{D,k}\, w_{k|k-1}^{(j)}\, q_k^{(j)}(\mathbf{z})}{\kappa_k(\mathbf{z}) + p_{D,k} \sum_{l=1}^{J_{k|k-1}} w_{k|k-1}^{(l)} q_k^{(l)}(\mathbf{z})}, \tag{5.21}$$

where

$$q_k^{(j)}(\mathbf{z}) = \mathcal{N}(\mathbf{z};\, \mathbf{H}_k \mathbf{m}_{k|k-1}^{(j)},\, \mathbf{H}_k \Sigma_{k|k-1}^{(j)} \mathbf{H}_k^T + \mathbf{R}_k), \tag{5.22}$$

and κ_k denotes the clutter measurement PHD. The weighting factors $w_k^{(j)}$ of the updated components, which are a function of the probability of detection $p_{D,k}$, are calculated using the likelihood of associations in the form of statistical distances $q_k^{(j)}(\mathbf{z})$. These statistical distances are determined by evaluating the innovation distributions, expressed by Equation 5.22, at a given measurement $\mathbf{z}$, as a measure of the association likelihood. For each given measurement, the association probabilities with each propagated Gaussian component are calculated. These association probabilities for a given measurement are summed, and the weighting factors associated with each one of the updates are a fraction of each individual association probability, over the total summed association probabilities for a given measurement, as expressed by Equation 5.21. Measurements that are closer to the innovation means $\mathbf{H}_k \mathbf{m}_{k|k-1}^{(j)}$ have larger association probabilities and hence result in larger individual weighting factors, since they constitute a majority of the total sum of association probabilities.

The MHT dynamics can be observed by inspection the update term in Equation 5.16, as expressed by Equation 5.17. For each measurement in the set $\mathbf{z} \in Z_k$, each propagated Gaussian component is updated using the standard KF update equations expressed by Equations 5.18 through 5.20. Equation 5.17, as a function of each $\mathbf{z}$, iteratively updates all the Gaussian components with each measurement.

All of these updated posterior components that form different estimates will continue to be propagated as priors for the next time step. Targets that were not detected are retained by the missed detection term, and since all potential associations are made, the correct data associations were made. Posterior components with larger weighting factors are more likely to be the correct associations. As incorrect associations continue to be propagated and updated, their weighting factors will continue to drop since the evaluation of their innovation at the newly obtained measurements will be small. Reduction in a weight factor is not only an indication of incorrect association, but in the PHD formulation it is the result of integrating over the PHD

mass region, which gives the expected value of the number of RFS elements in the region. Hence, this reduction signifies a reduction in the expected number of set elements within the region over which the relevant Gaussian component is distributed. The total expected number of elements in the set can be calculated by simply adding all of the weighting factors together, which is equal to integrating over the entire PHD mass.

Franken et al. [32] noted that in the event of a missed detection, a disproportional amount of the PHD mass shifts from the missed target to the detected targets, regardless of the distance between the two targets. This effect was coined as the *spooky effect*. Vo et al. [71] did an investigation and concluded that the effect is not a result of the RFS formulation, but the result of the first order statistical approximation. Vo et al. implemented a variant of the multi-target Bayes filter and demonstrated that the spooky effect is not present, and is only a factor once the PHD approximation is made. The effect is also present in the cardinalized PHD filter, and hence is not caused by the Poisson cardinality distribution assumption. Figure 5.1 depicts the PHD mass shifting from the missed target to the detected target, and the resultant disproportional mass in the region of the detected target.

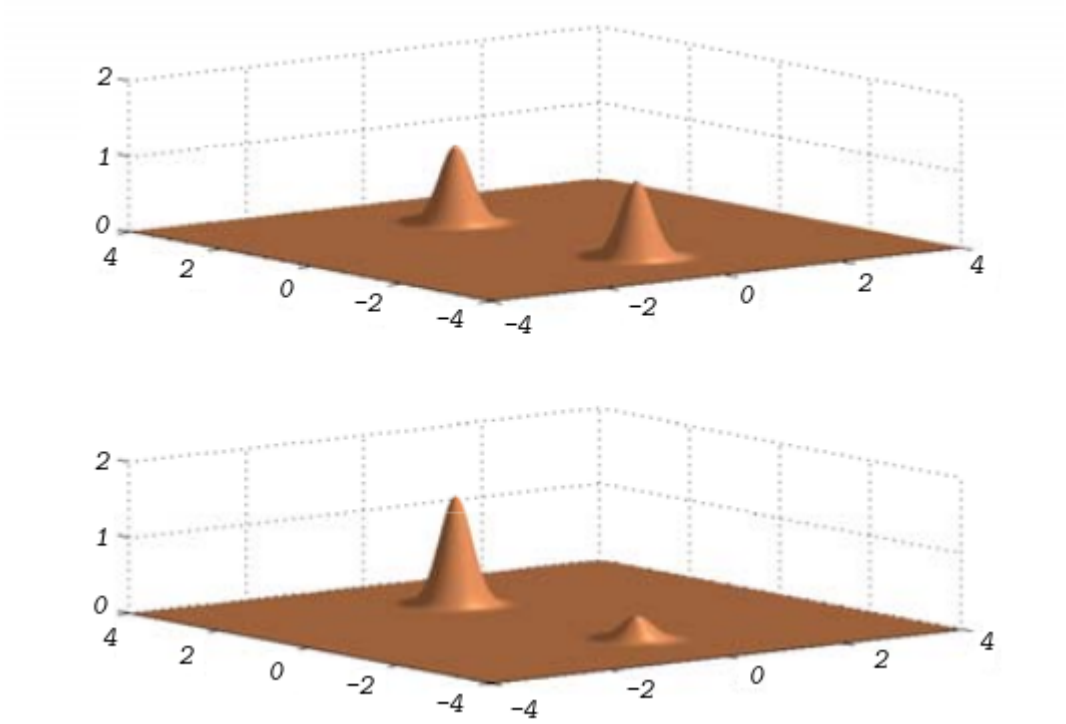


Figure 5.1: Illustration of the spooky effect. In the second image the right side target was not detected, resulting in a disproportionally large volume in the region of the left target. Image reproduced with permission from Vo [71].

5.3 Clutter PHD

$\kappa_k(\cdot)$ represents the PHD of the clutter RFS and is dependent on the expected number of clutter measurements and the distribution type. In Equation 5.21 the clutter is evaluated at the obtained measurement and reduces the resultant PHD weighting, that is its confidence.

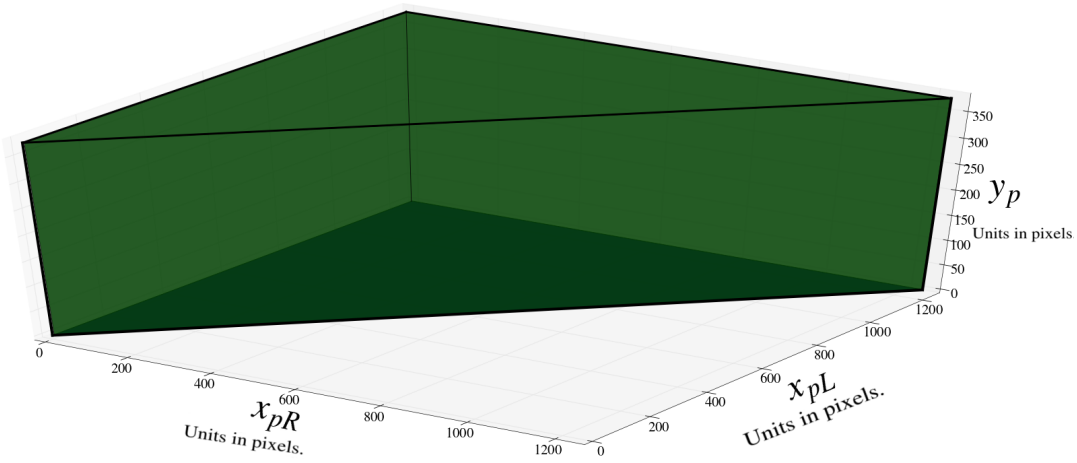


Figure 5.2: Visualisation of three dimensional measurement space. This space consists of all the possible measurement combinations given the positive disparity constraint.

A uniform distribution is used for the clutter PHD, expressed as

$$\kappa_k(\mathbf{z}) = \frac{\lambda_{ex}}{0.5 i_H i_W^2}, \tag{5.23}$$

given that clutter is assumed to have equal likelihood to appear anywhere. i_W represents the width of the image (maximum value of $x_{pR/L}$), i_H represents the height of the image (maximum value of y_p), and λ_{ex} the expected number of clutter measurements. Since the measurement space is three dimensional, the uniform distribution is distributed over a volume mass that represents all measurement combinations, which is depicted by Figure 5.2. Given the positive disparity constraint, x_{pL} has to be larger than x_{pR}, which causes the triangular shape in Figure 5.2. As a result, the volume can be represented by the denominator of Equation 5.23. The evaluation of the uniform distribution is the reciprocal of the volume. Combining the reciprocal of the volume and λ_{ex} together results in the used clutter PHD. Integrating over the entire region results in λ_{ex}, which corresponds to the definition of the PHD as the expected number of elements in the RFS.

5.4 Measurement and State Coordinate Descriptions

The expounded GM-PHD filter can be used along with matched features to result in state estimates of points. These points characterise the environment and the movement present in

the environment. Because of the robot's ego motion, each measurement is obtained within the camera's new coordinate system, and when triangulated into state space a position measurement is obtained. This measurement is described with respect to the current camera coordinates. Since the robot is moving through the environment, each measurement of a feature is described with respect to a different coordinate system. In order to relate state estimates, described with respect to different coordinates, to newly obtain measurements, described with respect to the current camera coordinates, a transform needs to be used. The transform, as derived and expressed in homogeneous coordinates in the literature review, can be expressed in Cartesian coordinates as

$$\mathbf{p}_C = \mathbf{R}_W \, \mathbf{p}_W + \mathbf{t}_W, \tag{5.24}$$

where $\mathbf{R}_W$ denotes an orthonormal rotation matrix and $\mathbf{t}_W$ a translation vector, describing the pose of an inertial axis in camera coordinates. This transform can be used to transform a point described in the inertial coordinate system $\mathbf{p}_W$ to a camera coordinate system $\mathbf{p}_C$. The reverse, from camera coordinates to inertial, can similarly be determined. In order to make use of these transforms, from inertial to camera and vice versa, the pose of the robot with respect to an inertial framework needs to be known. Inertial sensors such as an IMU can be used to obtain the robot's pose. Once the pose is known, state estimates can be transposed into camera coordinates and be updated with measurements obtained with respect to the current camera coordinates using the GM-PHD filter.

The pose of the robot has localisation error, and therefore uncertainty, since the pose is not known precisely but obtained with IMU measurements. IMU measurements have sensor noise that needs to be characterised. Figure 5.3 depicts a simple example of the results of pose uncertainty due to noise. Uncertainty in position and velocity moves the point cloud back and forth linearly, but uncertain orientation results in different directions of movement which causes the arc shape in the point cloud.

The nonlinear transform and resultant distribution is approximated with first-order Taylor expression (red ellipse) and the unscented transform (blue ellipse). Just like the nonlinear triangulation transform case, the unscented transform is a better approximation of the distribution then the Taylor approximation. This can be seen in Figure 5.3 given that the blue ellipse is more representative of the green ellipse than the red ellipse, the green ellipse being the spread of the resultant Monte Carlo samples.

The uncertainty present in the pose causes increased uncertainty in the transposed state estimates. The proposed algorithm transposes the resultant state estimates to inertial coordinates after the measurement update. Once new measurements are obtained, they are transposed into the new camera coordinates before the states are predicted and updated. The pose update (from

IMU measurements) and the update of the PHD Gaussian components are done separately, as opposed to jointly. Typically, in the field of probabilistic robotics, the pose and target state estimates are updated jointly in order to incorporate statistical dependency. However, it is not self-evident how a joint update could be incorporated in a PHD MHT framework. This topic is discussed further in the future work section.

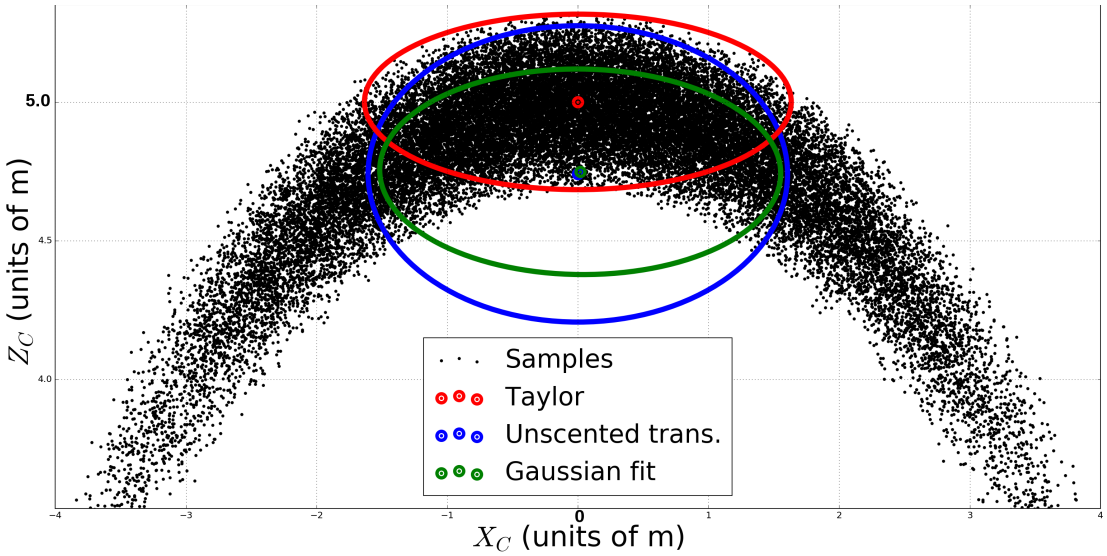


Figure 5.3: Translation and rotation transforms of two dimensional state space Monte Carlo samples. A robot moves from the origin to $(0,\ 5)$ with uncertainty in its pose. The influence of uncertainty in position, velocity, and orientation can be viewed. The red ellipse represents a Taylor approximation, the blue an unscented transform approximation, and the green represents the spread of the samples.

In this chapter the GM-PHD filter has been expounded, while in the previous chapter the creation of the measurement set is addressed. The strategy for transposing state estimates between coordinate descriptions is addressed, along with the influence of localisation uncertainty in the pose of the robot. The issue of an unbounded number of estimates due to the GM-PHD filter and the MHT framework is addressed in the next chapter.

6. MANAGING MULTIPLE HYPOTHESES

The MHT approach, as implemented in the PHD filter framework, results in state estimates for a time varying number of targets given the available information, and is capable of handling missed detections in the presence of clutter. However, the MHT approach results in an impractical implementation, due to an unbounded number of tracks (subsequent estimates). Figure 6.1 depicts the results of generating tracks from one state estimate from $k = 0$ to $k = 3$. Assigning all the new obtained measurements to each of the previously generated tracks result in an ever growing association tree. This process is further exacerbated by larger measurement sets and more initialised birth components. This unbounded association tree is especially computationally crippling for stereo vision applications due to the necessarily large size of the measurement set at each time step. Reducing the cardinality of the measurement set runs the risk of not covering a desired region with sufficient features, which results in poor representation of the environment and possibly missing moving objects completely.

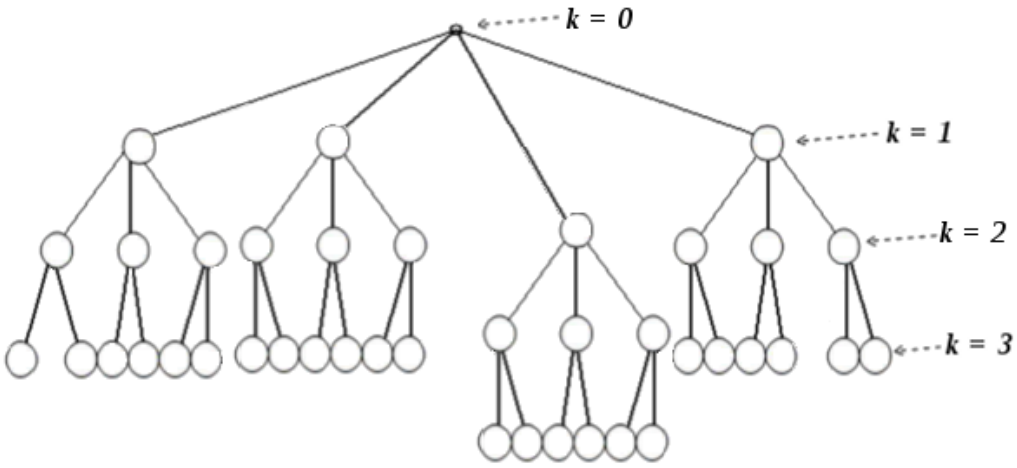


Figure 6.1: Illustration of an expanding track tree over 3 discrete instances as a result of the MHT data association strategy. 4 measurements are obtained at $k = 1$, 3 measurements are obtained at $k = 2$, and 2 measurements are obtained at $k = 3$. Image reproduced with permission from Bar-Shalom [15].

Given the unbounded growth of newly generated tracks, it is necessary for the purpose of implementation to remove some tracks (Gaussian components). It is also computationally

advantageous to remove or avoid any unnecessary new tracks. The sooner a track is cut off, the more computational savings are gained given the rapid growth. The metrics that can be used to inspect for viable augmentation or removal of tracks (branches in Figure 6.1) are the Gaussian weighting factors and the state estimates themselves. Three methods will be outlined to help with unbounded track growth, of which two are proposed by Vo and Ma's original GM-PHD paper [52], these are branch pruning and merging. The third is a gating strategy, which is a bit more unorthodox given the spirit of the approach with which the MHT framework resolves the data association problem. A fourth strategy, which is not used, would be simply limiting the total number of allowed Gaussian components. This limitation could be imposed by sorting the Gaussian PHD components in order of their weighting factors, and the top x selected and the rest discarded.

6.1 Branch Manipulation

This section will expounded the pruning and merging methods used to address the unbounded estimates problem. These methods augment existing tracks.

Pruning Branches

The first method is simple *pruning*. Branches of the tree are pruned (removed) if Gaussian PHD component weighting terms drop below a specified threshold. A weighting factor, in the context of the PHD framework, indicates the expected number of RFS elements in the region over which the Gaussian component is distributed. If a track was generated by an incorrect association, then any further associations would gradually cause the subsequent components' weighting factors to continue to drop, and approach zero. Once some of them drop below a threshold where they make up an insignificant portion of the PHD mass, they can be discarded, resulting in an approximated PHD mass function.

Merging Branches

The second method is *merging*. Branches that are sufficiently close to one another are merged together into one branch. The Mahalanobis distance metric is used to test whether two or more branches are within close proximity, proposed by Vo and Ma [52], described by

$$(\mathbf{m}_k^{(i)} - \mathbf{m}_k^{(\cdot)})^T \left(\Sigma_k^{(i)}\right)^{-1} (\mathbf{m}_k^{(i)} - \mathbf{m}_k^{(\cdot)}) \leq U_T, \tag{6.1}$$

where the Mahalanobis distance is squared. The means of all components $\mathbf{m}_k^{(\cdot)}$ are tested with respect to component i. All of the resulting distances that are smaller than the desired threshold U_T form a subset L_{set} of all the components that are to be merged. The weighting

factor of the new single Gaussian is the sum of the weights of the branches included in the set L_{set} as expressed by

$$\widetilde{w}_k = \sum_{i \in L_{set}} w_k^{(i)}. \tag{6.2}$$

As a result, the expected number of RFS elements are still the same, but are approximated in their localisation in the PHD mass. The new mean is a weighted average of all the neighbouring branches, using each Gaussian's weighting factor as expressed by

$$\widetilde{\mathbf{m}}_k = \frac{1}{\widetilde{w}_k} \sum_{i \in L_{set}} w_k^{(i)} \mathbf{m}_k^{(i)}. \tag{6.3}$$

The covariance is similarly calculated as the weighted average between the Mahalanobis used covariance, and the spread between the calculated average mean and the Mahalanobis used mean as expressed by

$$\widetilde{\Sigma}_k = \frac{1}{\widetilde{w}_k} \sum_{i \in L_{set}} w_k^{(i)} \left(\Sigma_k^{(i)} + (\widetilde{\mathbf{m}}_k - \mathbf{m}_k^{(i)})(\widetilde{\mathbf{m}}_k - \mathbf{m}_k^{(i)})^T\right). \tag{6.4}$$

If the dominant mass of multiple Gaussian components, constituting of the PHD mass, are distributed over the same region, then they can be approximated by being merged together and represented by a single Gaussian component. Merging reduces the amount of individual Gaussian components by approximating adjacent components with a single Gaussian component. The merging, being an approximation, is not an exact representation. Depending on the threshold value or the nature of the individual Gaussian distributions, the result could be less than desirable regarding its accuracy as an approximation.

6.2 Branch Avoidance

The third method uses validation gating, similar to the JPDA or GNN measurement validation process, resulting in a gated GM-PHD filter. The standard MHT strives to exhaustively associate all the measurements with a given target. However, given the probabilistic representation available as part of the PHD mass density, it could be argued that certain associations are unnecessary given sufficiently small association likelihoods. In the illustration represented by Figure 6.2, measurements 6 and 7 would be associated with the red and blue components in the full MHT (PHD) framework. However, it is clear given the validation regions that measurements 6 and 7 are unlikely to have been generated by the two targets. This argument assumes that the validation regions have been chosen to be reasonable representations of the covariances.

Gating Validation

The validation gating method attempts to find a middle ground between the GNN approach, which only assigns the most likely association hypotheses (for all targets), and the orthodox PHD filter, which considers all association hypotheses. This strategy was proposed by MacAgnano et al. [45] for a cardinalised PHD filter implementation. The gating method still approaches the problem with a MHT framework, but instead of generating tracks via all possible associations, associations are only made with validated measurements. The set of all possible associations for a given target is reduced to a subset of validated measurements by using the Mahalanobis distance. This set would consist of only the plausible association hypotheses, and would therefore reduce the number of unnecessarily associations and generated branches. This approach still avoids the undesirable risk of incorrect data association in the GNN framework, but alleviates some of the computational burden of the full PHD filter.

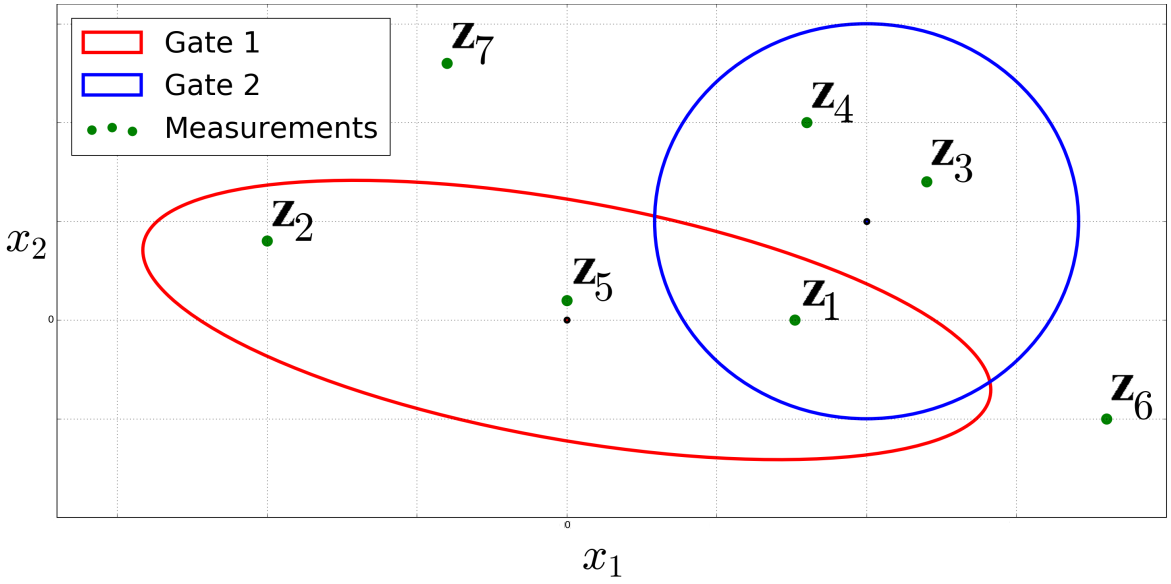


Figure 6.2: Two dimensional gating illustration. The red and blue ellipses represent validation gates while the green points are measurements.

The size of the validation regions need to be determined and could vary between different applications. Smaller regions result in more computational savings, but increase the risk that the desired measurements could be excluded from the validation regions. Larger regions result in less computational savings, but decreases the risk that the desired measurements could be excluded from the validation regions. It is more desirable to choose a conservatively larger validation region in order to ensure the correct association is made. However, the PHD's missed detection term would serve as a safeguard for correct associations that fall outside the validation region. In the event that only clutter is detected for multiple time steps, and the missed detected Gaussian component is truncated, the underlying target's estimate is reintroduced via

the birth components. Increasing the validation regions would increase the amount of validated clutter which would result in less computational savings. Even with extra undesired validated measurements, the gating strategy is still an improvement over the standard GM-PHD filter, given that those undesired validated measurements would have been associated with all targets anyway in the standard GM-PHD filter, as derived by Vo and Ma [52].

Once gating is used, the effective surveillance region is reduced from the total volume represented by Figure 5.2, to the collective three dimensional volumes encapsulated by all the validation gate bounds. The clutter PHD distribution is still assumed to be uniformly distributed, but is now distributed over these collective volumes. These volumes are calculated for each update stage. Each validation gate volume is expressed by

$$v_{gate} = c_M \sigma^{m_{dim}} |\mathbf{S}|^{0.5}. \tag{6.5}$$

$\mathbf{S}(k)$ is the innovation, g^M is number of standard deviations that set the boundary of the validation gate, and m_{dim} is the number of dimensions of the measurement space c_M is the volume of an unity hypersphere, that is $c_2 = \pi$ and $c_3 = \frac{4\pi}{3}$ [31].

Calculating the total volume of the gates introduces an issue of gating overlap. When the volumes of gates overlap with one another, it results in the overlapping region added to the total surveillance volume more than once, which results in an incorrect representation of the actual surveillance region volume. Using the example shown in Figure 6.2 with gating, the total surveillance volume would be the sum of the volumes (surface areas) of the two gates, which results in less volume than the entire measurement space. However, in this example the gating volumes overlap in the region of measurement 1 and causes that region's volume to be added one too many times.

The gating region, if large enough or near the boundaries of the measurement space, could also breach the boundary of the measurement space which would result in the addition of surveillance volume that is not even contained within the total measurement space. This issue is further exacerbated by more conservative validation gates. Any drastic method of testing for overlap between all the present Gaussian volumes would be computationally taxing, and unnecessary given the lack of actual gain from such a process. A simple test proposed by MacAgnano et al. [45] is expressed by

$$v_{used} = \min \{v_{gateTotal}, v_{all}\}. \tag{6.6}$$

If the overlap is pervasive to the degree that the surveillance region volume $v_{gateTotal}$ exceeds the total measurement space volume v_{all}, then the smaller volume is used.

A gating strategy makes use of the probabilistic description of state estimates in order to avoid making highly unlikely data-to-target associations. Avoiding these associations addresses the unbounded estimates issue and results in computational savings. Gating and merging strategies are used to approximate the PHD mass. The weightings are used to reject Gaussian components. Gaussian components that are within proximity to one another are approximated with a single distribution. These methods solve the issue of an unbounded number of estimates in the MHT framework.

The following chapter will address the implementation, testing, and evaluation of the proposed algorithm.

7. EXPERIMENTS AND RESULTS

This chapter outlines the proposed algorithm, the dataset used to test the algorithm, the methods used to analysing the results, and the results of the analysis. Section 7.1 presents the implementation of the various methods chosen and discussed in their respective chapters. Section 7.2 presents the test environment and dataset used to investigate the algorithm. Section 7.3 outlines two methods used for testing and analysing the results. Section 7.4 presents the results of the evaluation. The estimation accuracy, extent representation, computational time, and missed detections are investigated.

7.1 Implementation

The implemented algorithm consists of the feature detection methods, the state estimation method, limiting unbounded estimates methods, and transposing state estimate representations between coordinate systems. These methods are used in order to achieve the desired goal of robust DATMO. KAZE, A-KAZE, and ORB were implemented using OpenCV libraries. A maximum limit of 300 features for each stereo image was imposed. Detected features are matched between stereo rectified images. Feature had to pass the ratio test [44], which matches features both ways between images. Feature matches had to be strong enough, that is an upper limit on the distance metric. The matching process takes into account the positive disparity and stereo rectification constraints, resulting in reliable feature matches. Feature matches can be triangulated into the state space using the derived transform.

The obtained matches form the measurement set used in the PHD framework. The PHD filter is implemented using a GM density-based description. The implemented GM-filter results in six-dimensional point state Gaussian estimates, consisting of three-dimensional position and velocity $[x_k, \dot{x}_k, y_k, \dot{y}_k, z_k, \dot{z}_k]^T$, and assumes a constant velocity model as the motion model. The derived camera projection transform is used as the measurement model. The nonlinear camera projection transform is approximated with the unscented transform. New birth GM components are introduced every 0.5 seconds.

The unbounded estimate problem, present in the MHT framework, is addressed with pruning, merging, and measurement validation gating. The clutter is assumed to be uniformly

distributed over the surveillance region, which is the collective gated region. The pose of the robot, assumed to be Gaussian and approximated with the unscented transform, is estimated using accurate IMU measurements. The pose is used to transpose GM components between coordinate systems. After GM components have been updated, they are transposed into an inertial coordinate system. At the next time step, the state predictions and measurement associations are done after the GM components have been transposed into the current coordinate system, and after the pose estimate has been updated.

The cameras obtain measurements within the vehicle's body coordinate system. The IMU module is used to obtain the vehicle's pose with respect to the initial starting position, which serves as the inertial coordinate axis. The pose is used as a means of transposing all state estimate into a shared coordinate system in order to facilitate operations and Bayesian measurement updates.

7.2 Test Environment

The KITTI dataset was used in this thesis to test the developed algorithm [34]. Figure 7.1 illustrates the sensors used and their configuration. The KITTI dataset consists of an OXTS RT 3003 Inertial Measurement Unit (IMU) [10] yielding pose measurements, stereo camera pair with a baseline of about 0.5 m, and HDL-64E LiDAR laser-scanner [3], yielding point cloud LiDAR measurements. Cameras generate images at roughly 10 Hz and are triggered when the rotating velodyne scanner is facing forward. The dataset contains images that are undistorted, stereo rectified, and time synchronised, with the intrinsic parameters identified by calibration.

The IMU used is the OXTS RT 3003, which yields positioning accuracy of up to 1 cm using real time kinematic (RTK) satellite navigation, and 0.1^o orientation accuracy [34], [10], as displayed in Appendix A. LiDAR measurements from the KITTI dataset are used as an indication of the estimation accuracy. The HDL-64E LiDAR has usable returns from 0.92 m up to 120 m, with a distance accuracy of less than 5 cm [3], as shown in Appendix A. The LiDAR measurements are not used in the filtering process, but instead are used as ground truth for analysis of the results. The IMU of the KITTI dataset, measuring the pose of the camera coordinate axis, is used to estimate the pose of the robot.

Figure 7.2 is an image from the KITTI dataset with some of the LiDAR measurements, at a particular discrete time step, projected into the image plane. Figures 7.3 and 7.4 depict manual colour annotations of four moving objects in the environment. Figure 7.4 depicts a three dimensional plot of the same LiDAR measurements from Figure 7.2, with measurements behind the vehicle neglected. The annotated colours in Figures 7.3 and 7.4 correspond to one another.

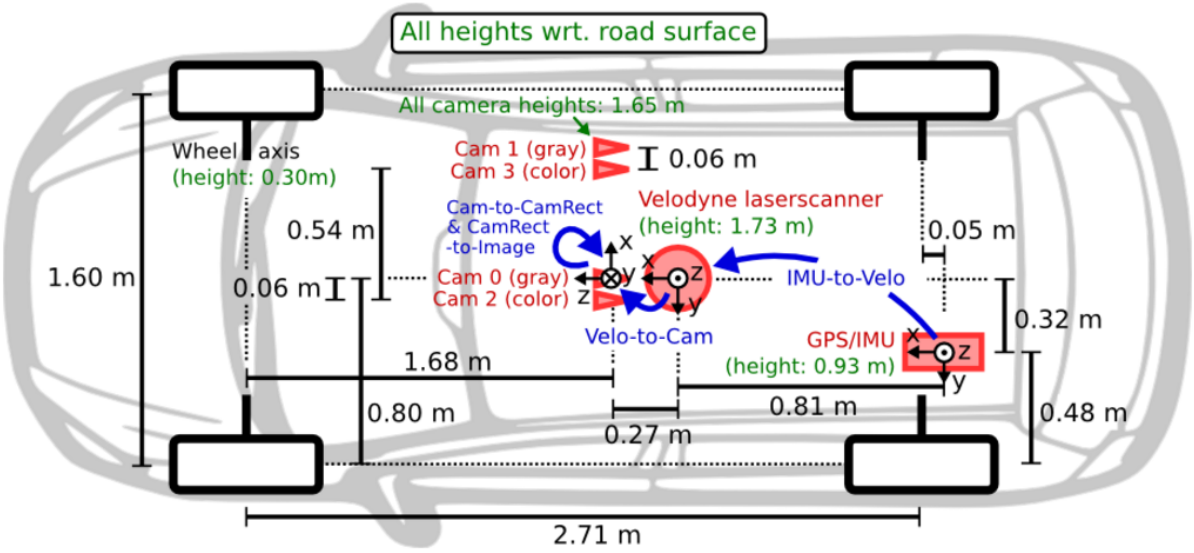


Figure 7.1: Top view graphic of the sensor configuration used to obtain the KITTI dataset. Image reproduced with permission from Geiger [34].

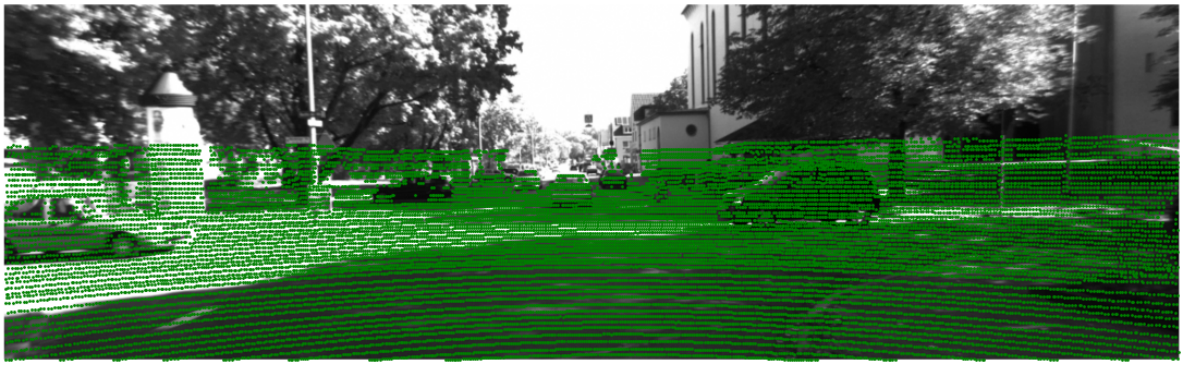


Figure 7.2: LiDAR measurements projected to left camera image plane.

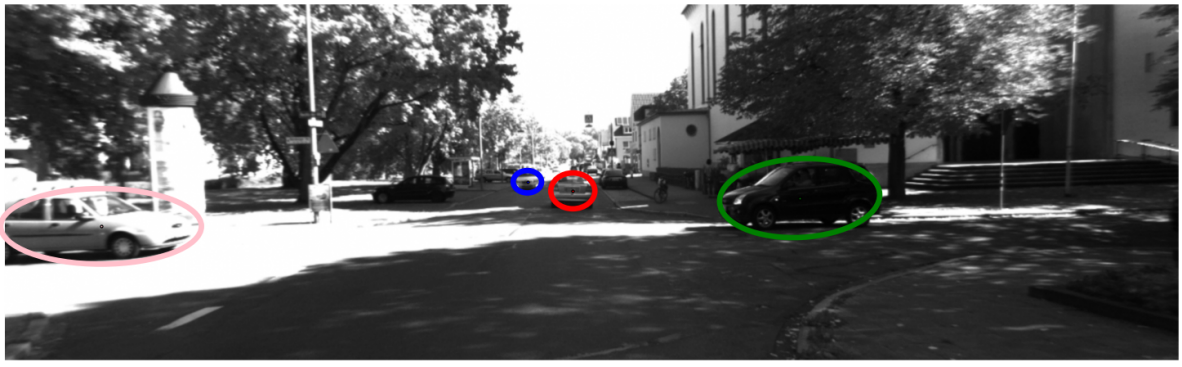


Figure 7.3: Four ellipse annotated moving objects in the scene using different colours. Ranked from close to distant they are: pink, green, red, and blue.

7.3 Methodology

Some concepts used to test the resultant Gaussian distributions are first introduced and explained. These concepts are, firstly, Kullback–Leibler (KL) divergence, and secondly, the Chi

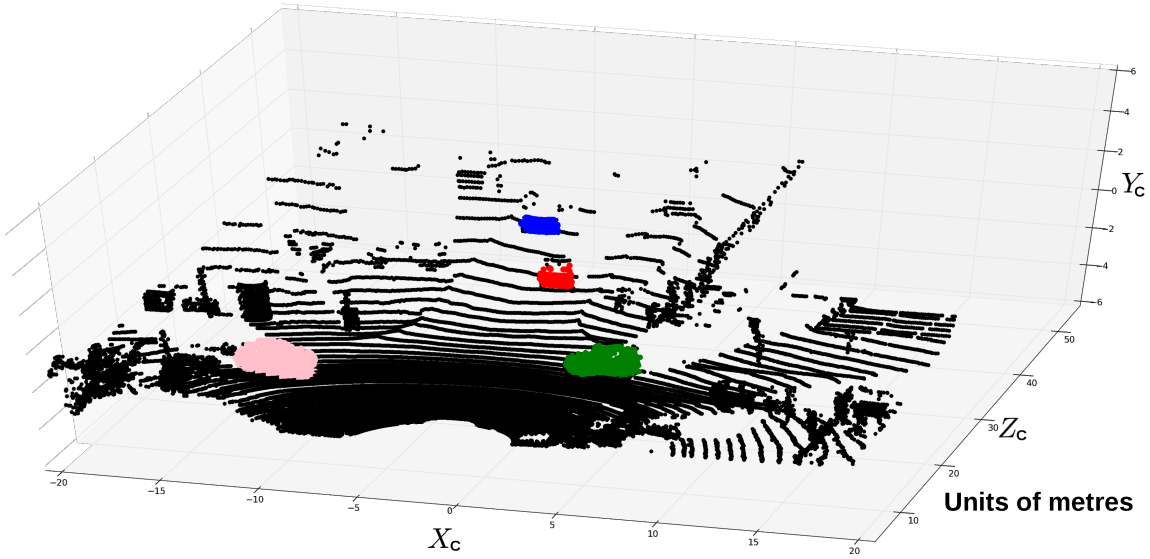


Figure 7.4: The same four colour annotated moving objects using LiDAR measurements. The colours correspond to the moving objects in Figure 7.3.

squared distribution as a distribution of the L2 norm of a standard Gaussian distribution. These methods were used to inspect the resultant distributions, specifically distribution accuracy and the spread of distributions in the environment. One dataset of manually annotated LiDAR measurements was used to inspect the spread of the resultant state space Gaussian distributions. The rest of the datasets, all unannotated, where used to test the accuracy of the Gaussian distributions.

Chi Squared Distribution

A Chi squared distribution $x \sim \chi^2(k)$, analytically expressed by

$$p_{\chi^2}(x;\, k) = \begin{cases} \frac{x^{(0.5k-1)} e^{-0.5x}}{2^{0.5k}\, \Gamma_{Gam}(0.5k)}, & x > 0 \\ 0, & \text{elsewhere} \end{cases}, \tag{7.1}$$

is a one-dimensional distribution on random variable x. A random variable defined as the L2 norm of a standard Gaussian distribution $\mathcal{N}(\mathbf{0}, I)$ is Chi squared $\chi^2(k)$ distributed, where $\Gamma_{Gam}(\cdot)$ denotes the Gamma function and I denotes the identity matrix. x is characterised by a single parameter, the degree of freedom parameter k, which is the number of dimensions of the standard Gaussian distribution. The Chi squared distribution is defined as the L2 norm distance, and therefore is a one-dimensional distribution. This distance corresponds to the square of the Mahalanobis distance, given that the Mahalanobis distance metric scales and normalises any Gaussian to a standard Gaussian. As a result, a squared Mahalanobis distance of a sample relative to any nonstandard Gaussian distribution would also be Chi squared distributed.

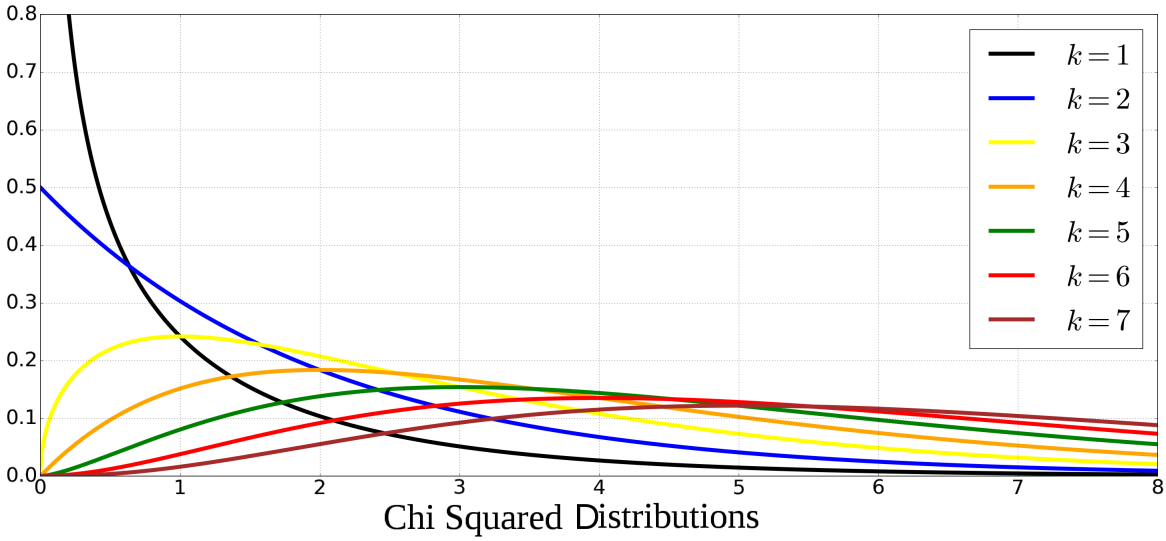


Figure 7.5: Chi squared distribution for various degrees of freedom.

Figure 7.5 depicts Chi squared distributions for a variety of degrees of freedom. The mean of the distribution is k and the variance $2k$. As the dimensionality of the Gaussian distribution increases, the probability mass of the Chi squared distribution χ^2 shifts to the right. As the dimensions increase, the number of random values that contribute to the Mahalanobis distance increases and therefore the distance increases. For example, if the distance is thought of as an Euclidean distance where the sample point vector is random with uncertainty that is statistically independent, then as the number of variables increase, the distance is likely to grow $\sqrt{(x)^2+(y)^2+\ldots}$ since each new dimension adds to the deviation.

Kullback–Leibler Divergence

KL divergence is a measure of the extent to which a distribution deviates from another distribution, specifically in quantities of information gain or loss (entropy), and is expressed by [53]

$$I_{KL}^{D}(A||B) = \sum_{i} A(i) \log \frac{A(i)}{B(i)} \tag{7.2}$$

for discrete random variables, and expressed by

$$I_{KL}^{C}(A||B) = \int_{-\infty}^{\infty} A(x) \log \frac{A(x)}{B(x)}\, dx \tag{7.3}$$

for continuous random variables. KL divergence is a nonnegative quantity. Zero KL divergence indicates that the distributions are the same, that is they do not diverge. As the two distributions differ, the KL divergence increases. KL divergence determines the information gained when distribution A is used instead of B. The quantity can also be thought of as expressing the information lost when distribution B is used to represent or approximate A, that is B's divergence from A. As a result, the function is not a symmetric function, $I_{KL}(A||B) \neq I_{KL}(B||A)$,

that is the information lost when B is used to approximate A is not the same as the information lost when A is used to approximate B.

7.4 Results

This section outlines how the methods mentioned in this chapter are used to evaluate the GM components that are obtained after implementing the proposed algorithm. The section regarding computation compares the computational duration of the proposed algorithm with various feature detection methods in a Python environment. Eleven datasets are evaluated. The section about distribution accuracy calculates the squared Mahalanobis distances of LiDAR measurements with respect to the GM components in order to test distribution accuracy. These distributions are expected to be Chi squared distributed χ^2. The section about point estimate spread determines the spread of the point estimates in order to quantify and determine whether the point estimates are collectively representative of the physical extent of moving objects. KL divergence from manually annotated extent distributions were used to quantify and compare results. This test was done on 1 dataset, given that data needs to be manually annotated, and the differences between ORB, KAZE, and A-KAZE were inspected. Missed detections were also inspected for the annotated dataset.

7.4.1 Distribution Accuracy

LiDAR measurements were used to inspect the accuracy of the resultant Gaussian distributions. Of particular interest is the accuracy of distributions along the camera projection lines due to the expected uncertainty along the camera projection lines. Figure 4.7 depicts a typical resultant state space distribution. The distribution is relatively certain in the dimensions that are perpendicular to the camera projection line, while quite uncertain in the dimension of the projection line. The dimension of the projection line points into the image as shown in Figure 7.6, and as a result only the parts of the GM components that are certain can be observed when projected into the image plane. The distribution is quite certain in the dimensions seen in the image plane, which correspond to the dimensions that are perpendicular to the camera projection line. The dimension not seen in Figure 7.6, but seen in Figure 4.7 is inspected for its accuracy, given the expected uncertainty, as a measure of reliability of the proposed algorithm for DATMO.

Figure 7.7 illustrates the approach. A LiDAR measurement that is sufficiently close to the camera projection of a Gaussian mean is used as ground truth for that Gaussian. The assumption is that the LiDAR measurement is sufficiently close to the mean, and therefore is a reasonable measurement. This assumption is based on the fact that the depth of an object does not vary dramatically on its observed surface, even though it does not exactly correspond to

the mean of the distribution. A LiDAR measurement is not used as ground truth if its image plane projected position is not within a two-pixel radius of the camera projected Gaussian mean. The LiDAR measurement, given its less than 5 cm accuracy (Appendix A), is assumed to be perfectly accurate.

Figure 7.6: Single confidence ellipse of an estimate distribution from gated GM-PHD filter projected into the right side image plane.

Once a corresponding measurement is found, the Mahalanobis distance of the LiDAR measurement with respect to the Gaussian distribution is determined in the state space along the projection line dimension, as M in Figure 7.7. However, the Mahalanobis distance is not determined in the full state space, but instead in one dimension, along the projection line dimension, since a hard correspondence has been made in the other two dimensions. This is represented by Figure 7.7. The distance M is determined in the dimension of the projection line, and is not the multidimensional distance directly from the measurement to the mean.

A unit vector is determined in the dimension of the projection line and a linear dot product transform of the full multi-dimensional distribution is determined. The resultant distribution (after the dot product operation) is a one-dimensional distribution of the state estimate Gaussian distribution over the projection line dimension. The Mahalanobis distance of the LiDAR measurement with respect to the one-dimensional distribution is calculated. Given that the gated GM-PHD distributions are Gaussian, the squares of the Mahalanobis distances should be Chi squared χ^2 distributed with one degree of freedom as depicted by Figure 7.5. The

squared Mahalanobis distances are determined over 30 seconds worth of data at 10 Hz. The results are displayed as normalised histograms of tightly discretised bins. The histograms, for different measurement detection methods, are plotted with the Chi squared $\chi^2(k=1)$ distribution for comparison, as seen in Figure 7.8. All the results, for all the datasets, are in Appendix B and are similar to the one depicted in this chapter.

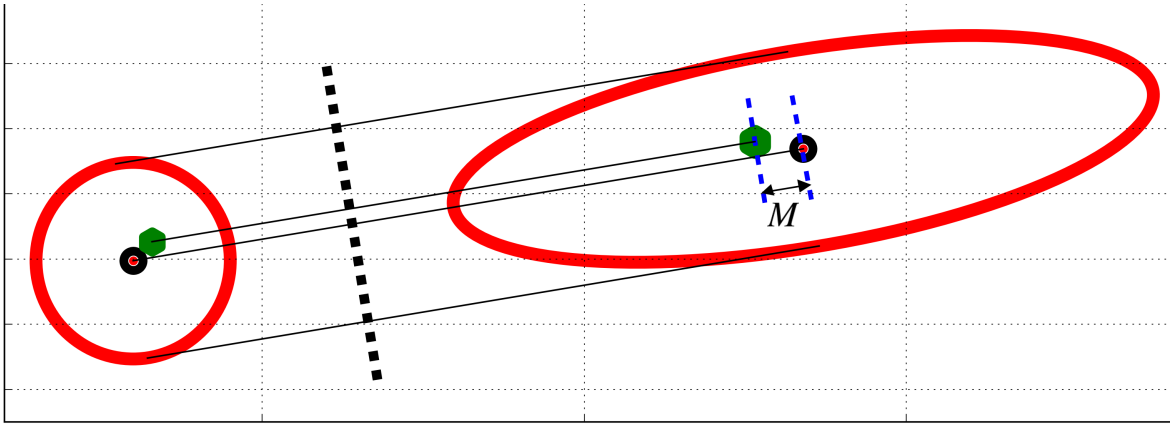


Figure 7.7: Simplified illustration of the accuracy testing method. On the left is the distribution as it appears in the image plane with a LiDAR measurement (green) as it appears in the image plane. On the right is the same distribution in the state space, with the same LiDAR measurement. An association has been made between the measurement and the distribution.

All the resultant histograms, relative to the Chi squared χ^2 distribution, are more uniformly distributed. The histograms, as expected, have large outliers toward the tail end of the distributions due to inaccuracies in LiDAR to Gaussian associations. This trend continuous for a large range of the distributions, but is cutoff by limiting the image axis at 8 in order to zoom in and inspect the distributions. Some large outliers are also expected given the nonlinear camera projection transform which does not result in perfect Gaussian distributions, but are instead approximated. The shift of the probability masses toward the higher end demonstrate that the estimated Gaussian distributions are more certain than one would expect them to be. This can be observed by comparing the histogram distributions to the Chi squared χ^2 distribution. Larger mass toward the lower end would, on the other hand, indicate larger uncertain than one would expect. This can be seen from the squared Mahalanobis distance expressed as

$$D^2_{MAL} = (\mathbf{x}_{LiDAR} - \boldsymbol{\mu})^T (\Sigma)^{-1} (\mathbf{x}_{LiDAR} - \boldsymbol{\mu}). \tag{7.4}$$

If the covariance Σ is reduced (increased certainty), then the Mahalanobis distance for the same measurement increases. The more uniform nature of the depicted histogram distributions is not unexpected, given the expected quadratic error of Gaussian estimates along camera projection lines. This error would result in larger Mahalanobis distances, and therefore more mass toward the higher end of the histogram distributions. The generated histograms yield a tangible metric

that can be used to adjust and test the effects of filter and feature detection parameters on the estimation accuracy. If certain variations cause the histograms to be more representative of the Chi squared distribution, then those variations result in a more accurate filter.

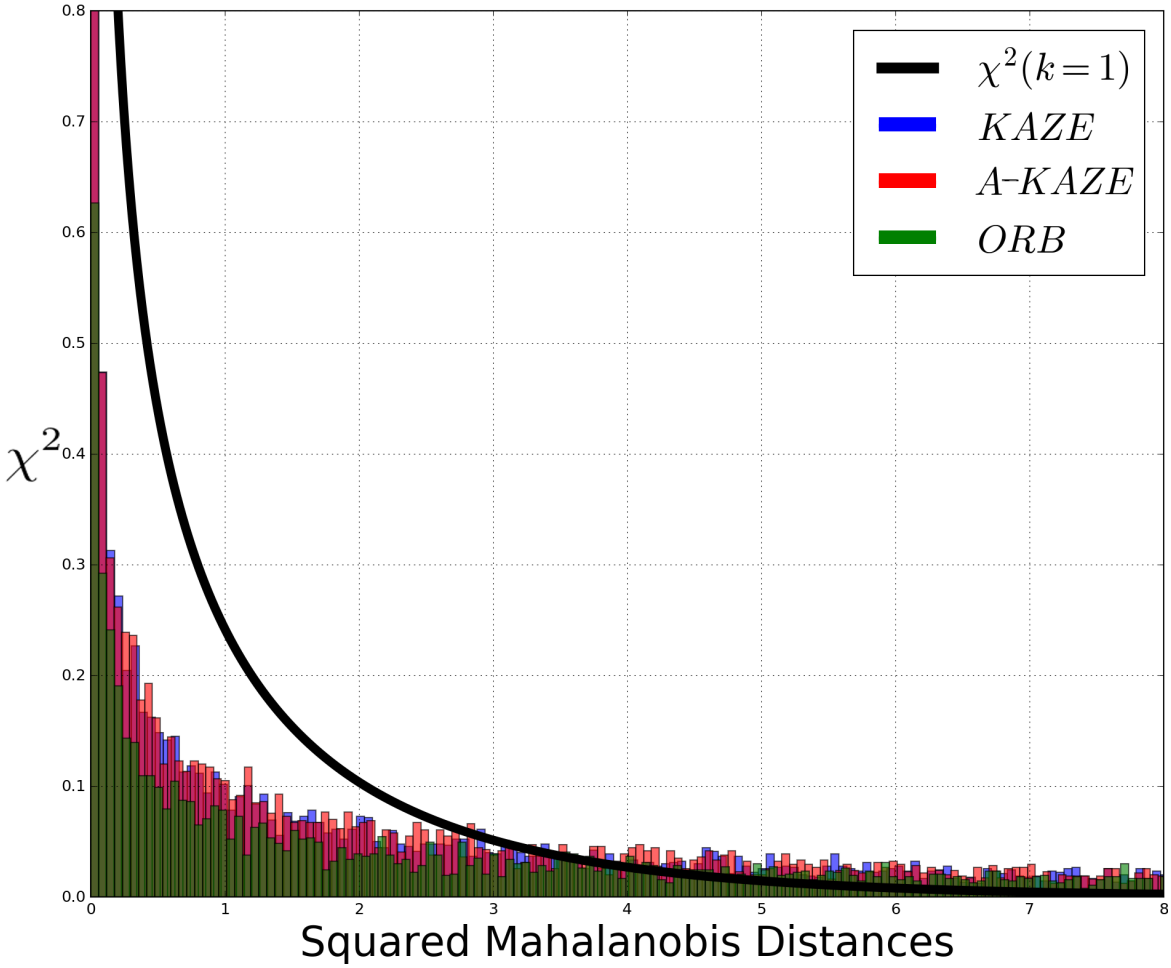


Figure 7.8: Chi squared χ^2 distribution plotted with the resultant histogram KAZE, A-KAZE, and ORB distributions. The histograms are created from Mahalanobis distance samples of LiDAR measurements with respect to the Gaussian estimates. The histograms are normalised and transparent. Bars are transparent, resulting in different colours once overlap occurs.

The distributions in Figure 7.8 also show that the probability mass of KAZE and A-KAZE are similarly distributed. This trend is observable in the rest of the results in Appendix B, with reasonable consistency. The ORB feature detection method results in a distribution that is more uniformly distributed both with respect to the Chi squared χ^2 distribution, and to the KAZE and A-KAZE distributions. Again, this trend is observable in the rest of the results in Appendix B, with reasonable consistency.

The reason for ORB's increased squared Mahalanobis distances, with respect to KAZE and A-KAZE, is most likely due to the lack of feature subpixel localisation accuracy. The ORB

feature detector algorithm, which is built on FAST, locates features at integer positions. Errors in the disparity $x_{pL} - x_{pR}$, caused by feature position inaccuracy, result in less accurate estimates. As stated before, these less accurate estimates are further exacerbated when the true distance of the detected objects are more distant with respect to the cameras. As the resultant Gaussian distributions are less accurate, the Mahalanobis distances become larger.

Given the observed trends, it would seem that KAZE and A-KAZE are more desirable feature detection algorithms. The observed results are expected given the lack of subpixel position accuracy of FAST, on which ORB is built.

7.4.2 Point Estimate Spread and Missed Detections

This subsection inspects the spread of the resultant GM components in order to determine the degree to which they represent the physical extent of objects in the environment. The frequency of missed detections are also investigated. The state model $[x, \dot{x}, y, \dot{y}, z, \dot{z}]^T$ used for the gated GM-PHD filter is a point state with 3 dimensions of position and 3 dimensions of velocity. The resultant Gaussian distributions are therefore point distributions. It is worth determining whether the resultant point distributions are sufficiently spread out, and therefore representative of the extent of moving objects. The extent of an object is important when evaluating the viability of an algorithm that tracks dynamic objects in an environment.

Point distributions could be quite densely situated on a moving object, but only on a part of an object, or on one side on an object, as seen in Figure 7.9. It is desirable that the distributions should be spread out over the surface of the object in order to be representative of the object's physical extent. The left side of the vehicle in Figure 7.9 does not have any point distributions. As a result, an algorithm that would use the resultant Gaussian point distributions would not be aware of the full extent of the vehicle.

The extent of moving objects were evaluated using KL divergence. The data of one dataset was manually annotated with extent, as seen in Figure 7.3, over 22 seconds worth of data. The extent ellipses represented in Figure 7.3 were used as the confidence ellipses of bivariate Gaussian distributions. These distributions will be referred to as the extent distributions $\mathcal{N}_{ex}(\boldsymbol{\mu}_{ex}, \Sigma_{ex})$. The KL divergences between the extent distributions and the spreads of the estimates on the objects were calculated. The spread was determined by

$$\Sigma_{spd} = \frac{1}{N} \sum_{i=1}^{N} (\mathbf{x}_i - \boldsymbol{\mu}_{spd})^2, \tag{7.5}$$

where $\mathbf{x}_i$ is the i-th mean of an estimate distribution projected into the image plane that is located on a specific moving object. N is the number of measurements on the object, and $\boldsymbol{\mu}_{spd}$ indicates the mean of these projected points. The resultant Gaussian distribution

$\mathcal{N}_{spd}(\boldsymbol{\mu}_{spd}, \Sigma_{spd})$ will be used as the measured extent, indicating the degree to which the measurements are spread out. The spread is depicted in Figure 7.9 as the largest red ellipse. The divergence would be from the green ellipse to the red ellipse, these representing the confidence ellipses of Gaussian distributions.

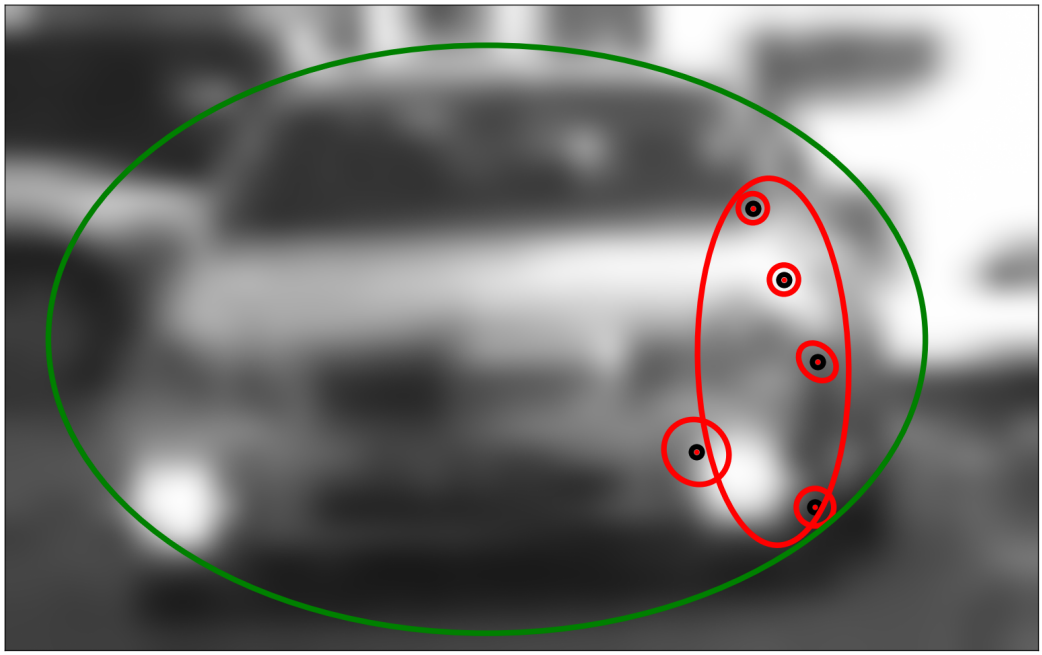


Figure 7.9: Illustration of the spread of distributions on a moving object in comparison with its extent. The green ellipse represents the extent of the vehicle. The smaller ellipses represent Gaussian state space distributions projected into the image plane. The single large red ellipse (without a centre point) is the spread of these distribution means.

The KL divergence from $\mathcal{N}_{ex}(\boldsymbol{\mu}_{ex}, \Sigma_{ex})$ to $\mathcal{N}_{spd}(\boldsymbol{\mu}_{spd}, \Sigma_{spd})$ was determined for all objects in the scene at every discrete time step. These KL divergences were added together over the entirety of the dataset time and averaged. This process was used for each feature detection method, KAZE, A-KAZE, and ORB. Given that a KL divergence of zero indicates no divergence $I^D_{KL}(A||A) = 0$, that is the two distributions are the same, smaller KL divergence results are desirable. Smaller divergences indicate that the spreads are closer to the annotated extents, which indicates a more desirable representation of the extent of moving objects. The KL divergence for a continuous Gaussian to another continuous Gaussian assumes the closed form expression [53]

$$D_{KL}(\mathcal{N}_{ex}||\mathcal{N}_{spd}) \triangleq I^{C}_{KL}(\mathcal{N}_{ex}||\mathcal{N}_{spd}) \tag{7.6}$$

$$= \frac{1}{2}\left(\text{Trace}\left(\Sigma^{-1}_{spd}\,\Sigma_{ex}\right) + (\boldsymbol{\mu}_{diff})^{T}(\Sigma_{spd})^{-1}(\boldsymbol{\mu}_{diff}) - k + \ln\left[\frac{|\Sigma_{spd}|}{|\Sigma_{ex}|}\right]\right), \tag{7.7}$$

in units of *nats*, where k is the number of dimensions, that is 2, and Trace$(\cdot)$ is the trace operation. The difference between the means is expressed by

$$\boldsymbol{\mu}_{diff} = \boldsymbol{\mu}_{spd} - \boldsymbol{\mu}_{ex}. \tag{7.8}$$

Some alterations were made for computational reasons. Determining the spread for only two points, even though the event is rare, results in a correlation coefficient of $\rho = 1$, as expressed by

$$\Sigma_{spd} = \begin{bmatrix} \sigma_x^2 & \sigma_x\sigma_y\rho \\ \sigma_x\sigma_y\rho & \sigma_y^2 \end{bmatrix}. \tag{7.9}$$

A correlation coefficient of $\rho = 1$ results in a zero determinant of the spread covariance Σ_{spd}. This is prevented for the sake of using Equation 7.7, which requires the calculation of the inverse of the spread covariance, which requires the determinant. This issue was address by enforcing a maximum limit of the absolute value of the correlation coefficient of 0.9. The spread for only one point, which is also rare, is simply a zero matrix. In the single point event a covariance with no correlation and with small diagonal values is assigned.

Table 7.1 displays the results for the three feature detection methods. The single or two point spread scenarios, for which computational interventions were necessary, though rare, result in disproportionally large KL divergences. Consequently, they heavy penalise the outcomes and are the chief contributors for the largest additions to the total KL divergence averages displayed in Table 7.1.

Table 7.1 clearly shows that KAZE performed the best, and therefore yields point mass distributions that are, on average, more characteristic of the physical extent of the different moving objects. The presented results are a function of the feature detector's underlying methods. ORB performs intensity-based checks, while KAZE and A-KAZE use gradients by inspecting the Hessian matrix response to find features. The results indicate that a gradient based approach yields better results. However, A-KAZE and ORB, not using the same method, resulted in similar performance. It is difficult to draw a general conclusion based on these results since the results are likely to be application specific. As discussed in Chapter 4, ORB tends to yield more features in gradient-less regions, such as the black vehicle in Figure 4.1. Since

vehicles tend to be relatively gradient-less, one would expect to find more ORB detected features on them. This, one would expect, would lead to more state distributions representing the vehicles in comparison with KAZE and A-KAZE, and would therefore result in larger extent distributions. Given this expectation, due to the lack of gradients on vehicles, the results are unexpected. Clustering these point estimates, resulting in state distributions with incorporated extent, would be the next step for the given results. The chapter on future work discusses this in more detail.

Table 7.1: The average KL divergences for each feature detection method.

	ORB	KAZE	A-KAZE
KL divergence	$29.12 * 10^3$	$8.24 * 10^3$	$28.11 * 10^3$

It is important to only interpret the results in Table 7.1 by considering the relative sizes of the average KL divergences. The KL divergence is being appropriated for the purpose of evaluating the spread of points. Therefore, the results should not be interpreted from the perspective of information theory, but instead as a measure of inverse similarity or overlap. A simpler overlapping of areas of ellipses could have been used, but would result in a more linear evaluation metric, although not exactly linear.

KL divergence penalises spreads of fewer points significantly heavier than a more linear alternative. Meaning that KL divergence penalizes significantly more for dropping from two points to one point, than it does for dropping from fifty to forty-nine. This is more desirable as an evaluation approach since the information (representation) gained about an object's extent is significantly larger when moving from one to two points, than moving from forty-nine to fifty points. This is due to the fact that forty-nine points, in most cases, is already a relatively extensive representation of the extent, where as one point is not.

Missed Detections

Table 7.2 displays the total number of missed detections. A missed detection, as it is used here, is the total number of times that zero state estimates were tracking a moving object. As a result, these numbers do not reflect the number of times that a measurement of a moving object was not obtained (the normal definition), since the PHD missed detection term would keep propagating some Gaussian components even if no measurement was obtained. The missed detections are the amount of times when moving objects were not represented by state estimate distributions at all. Unexpectedly, given its KL divergence performance, ORB had

the least number of missed detections, while KAZE and A-KAZE performed at about the same level. Interpreting the KL divergence results and the missed detection results together for ORB, would indicate that ORB frequently represents a given moving object with at least one state distribution, but not so many distributions as to desirably represent the extent of the moving object.

Table 7.2: Missed detections of annotated moving objects for each feature detection method.

	ORB	KAZE	A-KAZE
Missed detections	18	53	59
Missed detection rate	5.86%	17.26%	19.21%

7.4.3 Computational Cost

Table 7.3 displays the computational cost of the proposed algorithm for 11 KITTI datasets [34]. The datasets each have 30 seconds worth of data, with measurements at 10 Hz, resulting in 300 stereo images for each of the 11 datasets.

Table 7.3: Comparison of the proposed algorithm's computational duration for 11 datasets using different feature detectors. The largest computational duration per dataset is highlighted in bold. The smallest computational duration per dataset is underlined. Quantities are in units of minutes.

Dataset	ORB (in min)	KAZE (in min)	A-KAZE (in min)
0	241.02	**262.3**	128.02
1	**143.6**	82.4	25.75
2	**231.77**	129.63	49.45
3	**231.93**	100.22	29.55
4	**150.27**	129.73	43.12
5	**236.62**	181.47	55.38
6	**248.52**	142.02	39.9
7	**308.73**	230.92	86.18
8	192.1	**243.2**	88.47
9	**235.6**	101.93	24.32
10	**192.83**	186.37	34.9

The average computational times of the algorithm reflected in Table 7.3 is much too long to be useful for practical purposes, especially given the fact that the datasets are of 30 seconds worth of real-time data. The shortest execution time is 24.32 minutes, almost half an hour, which translates to roughly 48.64 seconds of computation for 1 second of data at 10 Hz. The computational results, despite the fact that the algorithm was implemented in a Python environment, is computationally intractable for real time implementation. Considering practical implementation would almost certainly require extensive parallel processing. Feature detection, Gaussian distribution propagation, and measurement updates could all be parallelised. A parallel process would still be limited by the algorithm itself as certain points. For example, in the way that the total association probability needs to be calculated before any weighting can be specifically assigned, given that each weight is determined by dividing by the total weight in Equation 5.21.

The A-KAZE feature detection method on average outperformed KAZE as expected. The ORB feature detection method on average, unexpectedly, was computationally more intensive than even KAZE. Given the faster binary sequence descriptors of ORB, and the faster FAST based interest point detector, one would expect ORB to outperform the vector descriptor and nonlinear scale space based KAZE. Especially given the extra computation required to construct the nonlinear scale space. However, ORB does construct a Gaussian scale space which takes time, but one would still expect the discretisation of a nonlinear scale space flow function to take longer. A-KAZE significantly outperformed all other methods with the smallest computational duration per dataset. The relative duration of A-KAZE with respect to the largest computational duration was as low as 10.32% for dataset 9, roughly 10 times faster. A-KAZE is clearly the more desirable method as computation is concerned.

Clearly A-KAZE is the most desirable given the timing results, but even A-KAZE's timing results do not seem to be as practical as one would desire them to be. Further investigation would need to be done in order to determine the viability of practical implementation.

Table 7.4 displays the results of three tests. The computational time spent on different functionality is displayed as a percentage of the total computational duration. The filtering process takes a large portion of the total computation as expected; roughly a third. However, the techniques employed to address the unbounded estimates problem, unexpectedly, takes the largest portion, at more than a half.

Of these techniques, the merging process by far takes the most time, since pruning merely tests each weight once. The merging process is complex given that the set of Gaussian distributions I_{set} inspected for merging is traversed, starting at the largest weighting and remove from the set as components are inspected, until the set is empty. A summary of the mering process is given by

given: GM parameters $\sim \{w_k^{(i)}, \mathbf{m}_k^{(i)}, \Sigma_k^{(i)}\}_{i=1}^{J_k}$, and a merging threshold U_T.
Set $l = 0$, and $I_{set} = \{i = 1, ..., J_k\}$.
repeat
$l := l + 1$
$j := \arg\max\, w_k^{(i)}$
$$L_{set} := \left\{ i \in I_{set} \mid (\mathbf{m}_k^{(i)} - \mathbf{m}_k^{(j)})^T (\Sigma_k^{(i)})^{-1} (\mathbf{m}_k^{(i)} - \mathbf{m}_k^{(j)}) \leq U_T \right\}$$
$$\widetilde{w}_k^{(l)} = \sum_{i \in L_{set}} w_k^{(i)}$$
$$\widetilde{\mathbf{m}}_k^{(l)} = \frac{1}{\widetilde{w}_k^{(l)}} \sum_{i \in L_{set}} w_k^{(i)} \mathbf{m}_k^{(i)}$$
$$\widetilde{\Sigma}_k = \frac{1}{\widetilde{w}_k^{(l)}} \sum_{i \in L_{set}} w_k^{(i)} \left(\Sigma_k^{(i)} + (\widetilde{\mathbf{m}}_k - \mathbf{m}_k^{(i)})(\widetilde{\mathbf{m}}_k - \mathbf{m}_k^{(i)})^T \right)$$
$I_{set} := I_{set} \backslash L_{set}$
until $I_{set} = \emptyset$.

Table 7.4: Table of timing allocation. **Filtering** denotes the time spent on the GM-PHD filter, that is state predictions, measurement updates, introducing spawn components, and also gating. **Feature detection** denotes the time spent detection and matching features. **Pose functions** denotes the time spent updating the robot's pose estimate and the time spent transposing Gaussian components between coordinate systems, using the robot pose estimate. **GM alterations** denotes the time spent on altering association branches, such as pruning and merging, but not gating.

Test #	Filtering	Feature detection	Pose functions	GM alterations	Total
1	30.96%	0.08%	6.38%	62.57%	99.99%
2	30.13%	2.48%	5.71%	61.67%	99.99%
3	36.35%	1.26%	8.01%	54.37%	99.99%

The proportion of the merging strategy is unexpected, and contributes to the majority of the total computational duration of the algorithm. Comparative tests would need to be done in order to determine the algorithm's cost with and without the merging process, if it were to be implemented on a GPU. Tests could also be done to determine the computational cost as a function of the merging threshold U_T.

8. CONCLUSIONS

8.1 Summary

This chapter contains a concluding overview and discussion of the results. The thesis inspects different approaches and methodologies that can be used for the purpose of DATMO in a robot's environment using stereo vision cameras. Of particular interest is the goal of robustly and reliably tracking moving objects with methods that model the certainty of the robot's representation of moving objects. Modelling the robot's certainty of the states of moving objects allows for a measure of their reliability. An ideal system needs to be capable of handling non-linearities and be robust to measurement noise, false measurements, and missed detections.

The pinhole camera model is outlined, justified, and serves as the basis of the mathematical description of the camera projection transform. The pinhole model is described by intrinsic camera parameters. The camera projection transform describes the measurement model used to transform a point in the 3D state space to a 2D point in an image plane along a camera projection line. Stereo image geometry is derived and expounded, and images are stereo rectified resulting in a horizontal epipolar constraint. The constraint results in a 3D matched feature pair instead of a 4D matched feature pair, since the individual features have the same vertical position in their image planes given the horizontal epipolar constraint. The stereo geometry describes how a 3D point transforms to a 3D matched feature pair. Stereo rectification is used in the feature matching stage to conveniently search for viable features matches in order to create the measurement set. This searched is constrained by the horizontal epipolar constraint and the positive disparity constraint.

With the derived geometry, an image feature-based approach was used in order to make use of the vision-based camera measurements. Features were chosen as opposed to flow methods due to the complexity of compensating for the relative flow caused by ego motion. Feature were detected and matched. The resultant matches form the measurement set used in the filtering application. Feature-based methods were evaluated and their underlying methods expounded. ORB, KAZE, and A-KAZE were chosen for comparison. ORB was chosen for its unique integer pixel location approach. A-KAZE was chosen for comparison with ORB, since both use binary sequence descriptors. KAZE was chosen to introduce a vector-based

descriptor among the binary sequence descriptors. KAZE is also of particular interest due to the nonlinear scale space method used as opposed to the typical Gaussian scale space used by SIFT and SURF.

Features were extracted and limited to a maximum of 300 per image. Relevant image detection algorithm parameters were chosen to reject weak responses. Feature detection algorithms were implemented using OpenCV libraries. Features were matched between stereo images. These matches had to meet certain criteria in order to be reliable and therefore result in a robust application. The matching distance metric had to be sufficiently small in order to be deemed reliable. Feature matches had to pass the ratio test and therefore match both ways between images, only matches that are found both when starting with the left camera's set and matching to the right, and when starting with the right camera's set and matching to the left, were used. The matching process was governed by the positive disparity constraint and the horizontal epipolar line constraint.

Feature matches, matched using the outlined constrains, form the measurement set that is used in state estimation. Matches are triangulated from the measurement space to the state space using the derived stereo geometry. However, this transform is nonlinear and needs to be implemented by a nonlinear approximation technique. The unscented transform was used as the approximation technique, given that it was deemed more accurate than the first-order Taylor approximation. The unscented transform is outlined along with the basis of probabilistic filtering, the Bayes filter. State estimates and measurements are modelled using random variables. Random variable modelling allows for a measure of the certainty of the state vectors of points. These points describe moving objects in the environment, but neglects their extent. In the relevant section the probabilistic filtering scheme was introduced, derived, and justified.

The unknown correspondence or data association problem that arises from a MTT framework is introduced and described. The three most popular techniques, GNN, JPDA, and MHT, their methods, their advantages and disadvantages, are expounded and discussed. JPDA was rejected due to its combinatorial complexity. GNN was rejected because it does not serve the ultimate goal of yielding a robust algorithm. MHT was chosen for its robustness, due to its exhaustive association philosophy. The MHT framework was expressed as the multi-target Bayes filter which models state estimate collections as RFSs. MHT was ultimately implemented in the multi-Bayes filter first order expectation approximation known as the PHD filter. The approximation relieved the joint complexity of the multi-target Bayes filter and resulted in integrals on the single target state space. The RFS intensities (PHDs) were modelled with Gaussian mixtures which results in simple Kalman filter-based prediction-update equations.

The impractical unbounded number of association tracks caused by MHT was addressed with various techniques. These techniques involved either altering existing association tracks or

preventing associations. The alterations were proposed by the authors of the GM-PHD filter. Implausible associations were prevented by using a gating strategy. The MHT framework was still retained, but exhaustive associations were only made with a subset of validated measurements. Validation testing was done using the Mahalanobis distance metric. The issue of reducing the surveillance region, the total measurement space, due to gating was addressed. The robot's pose, also modelled probabilistically, was used to transpose distributions between different coordinate systems, given that new measurements are obtained relative to different coordinate systems.

Results

Algorithms were tested using the KITTI dataset. The results were evaluated and analysed for its computational duration, accuracy, extent representation, and missed detections. The Mahalanobis distance metric and Chi squared distribution were used to evaluate the accuracy of the GM state estimates along the dimension of the largest expected error. The KL divergence was used along with a manually annotated dataset in order to determine the degree to which the physical extent of moving objects are represented. These tests were done using LiDAR measurements from the KITTI dataset.

The results of the accuracy testing demonstrated that KAZE and A-KAZE performed at about the same level, with ORB being the least accurate. ORB's performance with respect to KAZE and A-KAZE is most likely due to its lack of subpixel localisation that causes disparity error. This error results in larger Mahalanobis distances and therefore a disproportional amount of distribution mass at higher values. The results of the accuracy testing were expected given the methods underlying each method as outlined in the literature review.

The results of the extent and missed detection testing demonstrated that KAZE results in the best spread of point mass estimates, while ORB results in the least number of missed detections. The results would indicate, even though it might be circumstantial, that ORB as a feature detection technique is more likely to detect a given target, but will result in poor representation of the detected target's extent. This is likely due to its intensity-based testing, as opposed to gradient based testing. The relative similar performance of KAZE and A-KAZE regarding missed detects is expected, while the relative dissimilar performance of KAZE and A-KAZE regarding extent is not expected.

The results of the computational testing demonstrated that A-KAZE significantly outperformed KAZE and ORB for every dataset. ORB, unexpectedly, resulted in the largest computational cost. This is unexpected due to the speed of FAST on which ORB is built. In all likelihood, the relative speed of KAZE and A-KAZE in comparison with ORB probably speaks more to their efficiency, rather than it does to ORB's inefficiency. The GM merging

technique cost the most computationally. This was unexpected, but inspecting the complexity of the merging process reveals as much. The most undesirable of the results, including the other evaluation metrics, is the overall computational duration of the developed algorithm. Even when only considering A-KAZE, the faster method, the computational cost is still far too burdensome for practical implementation, regardless of the fact that the algorithm was implemented in a Python environment. Any practical implementation would require extensive parallel processing with timing schemes between hardware components. Some of the algorithm parameters, both feature detection and gated GM-PHD filter, would need to be adjusted by lowering the conservative standard that was imposed in order to result in reliable and robust DATMO.

Closing Remarks

Ultimately, implementing MHT with stereo vision, or any camera-based configuration, leads to issues, especially if the goal is a robust and reliable system. The strategies used in order to achieve a robust algorithm, such as MHT itself among others, demand rigorous and computationally taxing methods. Even when external techniques, like gating, are used, the resultant algorithm is still too costly. Vision-based sensors such as cameras yield a wealth of information about the environment of a robot. However, in order to extract the information in the best workable fashion is not self-evident. Feature-based techniques prove to necessarily result in a large amount of measurements. If the available information is to be used in such a way as to result in a reliable implementation, then it requires that many features be extracted. Many features need to be extracted in order to represent the environment reliably and extensively. This goal results in large measurement sets, which, when combined with techniques such as MHT, results in complex and impractical implementation. Reliable and robust filtering algorithms are necessarily computationally complex and therefore sensitive to the number of measurements, and cameras yield large measurement sets. Philosophically these two frameworks would seem to be at odds with each other if the goal is practical implementation, and the results demonstrate as much.

8.2 Contributions

This sections outlines some of the important contributions of the conducted thesis work.

1. The thesis provides an extensive overview of the different methods that can be used for DATMO using stereo vision cameras. Specifically, with robustness and reliability as design objectives. The derivation of the camera model, analysis of image process techniques, analysis of data association techniques, and the derivation of the random finite set filtering context are presented.

2. The popular feature detectors are outlined in detail to show the different underlying methods and how they contribute to the resultant feature detectors. These methods are SIFT, SURF, FAST, ORB, KAZE and A-KAZE. Various assumptions made for computational gain are shown. The different ways that feature points are detected are presented. Of interest are the new nonlinear scale space based feature detectors, as opposed to the popular linear scale space methods. How these methods contribute to feature reliability and accuracy are demonstrated.

3. The data association, or unknown measurement correspondence, problem that is present in a MTT scenario is explained. The three most popular methods used to address the problem are explained in detail in order to demonstrate their advantageous and disadvantageous. These methods are GNN, JPDA (PDA), and MHT. The thesis demonstrates how MHT, expressed in a Bayesian filtering context as the multi-target Bayes filter, is more robust than its competitors. These methods are compared given the issues of unknown correspondence, missed detections, time varying number of targets, and measurement clutter.

4. The multi-target Bayes filter, and its RFS formulation, along with assumptions made to alleviate computational complexity, are shown. These assumptions include both the assumptions made to reduce the multi-target Bayes filter to a first-order density-based filter (GM-PHD), and the assumptions employed in order to justify the gating strategy that was used.

5. The thesis demonstrates how there seems to be tension between using cameras as sensors with the design objective of producing a robust and reliable state estimation filter. The design objective necessarily results in a computationally complex state estimation algorithm that is sensitive to the number of measurements. Using cameras necessarily, given the design objective, results in a large number of measurements. This issue is problematic despite the gating strategy that was used.

8.3 Future Work

This chapter contains a short discussion of some of the potential future work that could build on the presented work.

• Given the computational issues arising from MHT in conjunction with camera sensors, an investigation would need to be conducted to evaluate the viability of practical implementation. Implementing the proposed algorithm on a GPU would require extensive parallel processing and multi-threading. A pipeline would need to be created between feature detection, GM-PHD equations, and transposing the resultant Gaussians to an inertial coordinate framework. The feature detection algorithm could be used in a parallel fashion on different image segments. The GM-PHD filter, given its multi-branch predication-update equations, offers many opportunities for parallelisation, especially for the propagation and update equations. The nature in which weightings are determined, proportional to the total weighting, could cause a bottleneck in the computations.

• The histogram distributions, that were expected to be Chi squared χ^2 distributed, offers a way to gauge the accuracy of the GM distributions as a function of algorithm parameters. These histogram distribution can be used as a measure of the accuracy of the GM distributions, given that they are a function of the various parameters of the image detection and GM-PHD filter algorithms. As the parameters are varied, their effects on the accuracy of the distribution can be observed by inspecting how the histogram's probability mass shifts.

• The proposed algorithm yields point estimates in the state space. These point mass estimates can be clustered together in order to represent a moving target and its extent. Points can be clustered based on similar movement and proximity. State vectors can be expanded to include extent information. A target can be represented by a centre point position and velocity, and a Gaussian approximated extent. These formulations can be expressed in an inverse Wishart GM-PHD (IW-GM-PHD) filter framework [30].

• The state vectors can be expanded to include representation of the appearance of a target. The physical appearance (colour, pattern) is used in the feature detection and feature matching processes, but neglected in the actual state representation. Expanding the states to include the appearance would allow for further complexity and innovation in association likelihood calculations. The weighting factors could be influenced by the physical appearance of targets, not only the position and velocity. The association would need to be determined for complex situations, for example if something is close in proximity but looks very different. Assumptions would need to be evaluated, such as the degree (or rate) to which the physical appearance can change between consecutive measurements.

• A somewhat unorthodox approach would be to try and incorporate a joint estimation of point states and pose into the PHD framework. The individual point target state estimates are estimated separately due to the first order statistical moment of the multi-target Bayes filter that removes the combinational complexity. In order to determine pose and state estimates jointly would require the expansion of state vectors to include robot pose. However, the result would be that each target state estimate will contain a different estimate of the pose, which was jointly determined with each target's individual motion. Consequently, this result would be interpreted as each moving target having a different representation and understanding of where the robot is located. The effects of this dissonance between the different target pose representations are unclear. All of these pose estimates could be added together in a Bayesian update fashion to yield a single pose belief, resolving the dissonance between representations, but again the results of such an approach is unclear. This approach would theoretically result in more accurate pose and state estimates since it would incorporate the statistical dependencies between targets and robot pose.

APPENDIX A

This Appendix contains some information from the data sheet of the HDL-64E LiDAR and OXTS RT 3003 IMU used by the KITTI dataset [34], [3], [10].

HDL-64E LiDAR [3]:

Sensor:	• 64 lasers/detectors • 360 degree field of view (azimuth) • 0.09 degree angular resolution (azimuth) • 26.8 degree vertical field of view (elevation) -+2° up to -24.8° down with 64 equally spaced angular subdivisions (approximately 0.4°) • <5 cm distance accuracy • 5-15 Hz rotation rate update (user selectable) • 50 meter range for pavement (~0.10 reflectivity) • 120 meter range for cars and foliage (~0.80 reflectivity) • >1M points per second • <0.05 milliseconds latency
Laser:	• Class lm - eye safe • 4 x 16 laser block assemblies • 905 nm wavelength • 10 nanosecond pulse • Adaptive power system for minimizing saturation and blinding
Mechanical:	• 12V input (16V max) @ 4 amps • <29 lbs. • 10" tall cylinder of 8" OD radius • 300 RPM - 900 RPM spin rate (user selectable)
Output:	• 100 MBPS UDP Ethernet packets

OXTS RT3003 IMU [10]:

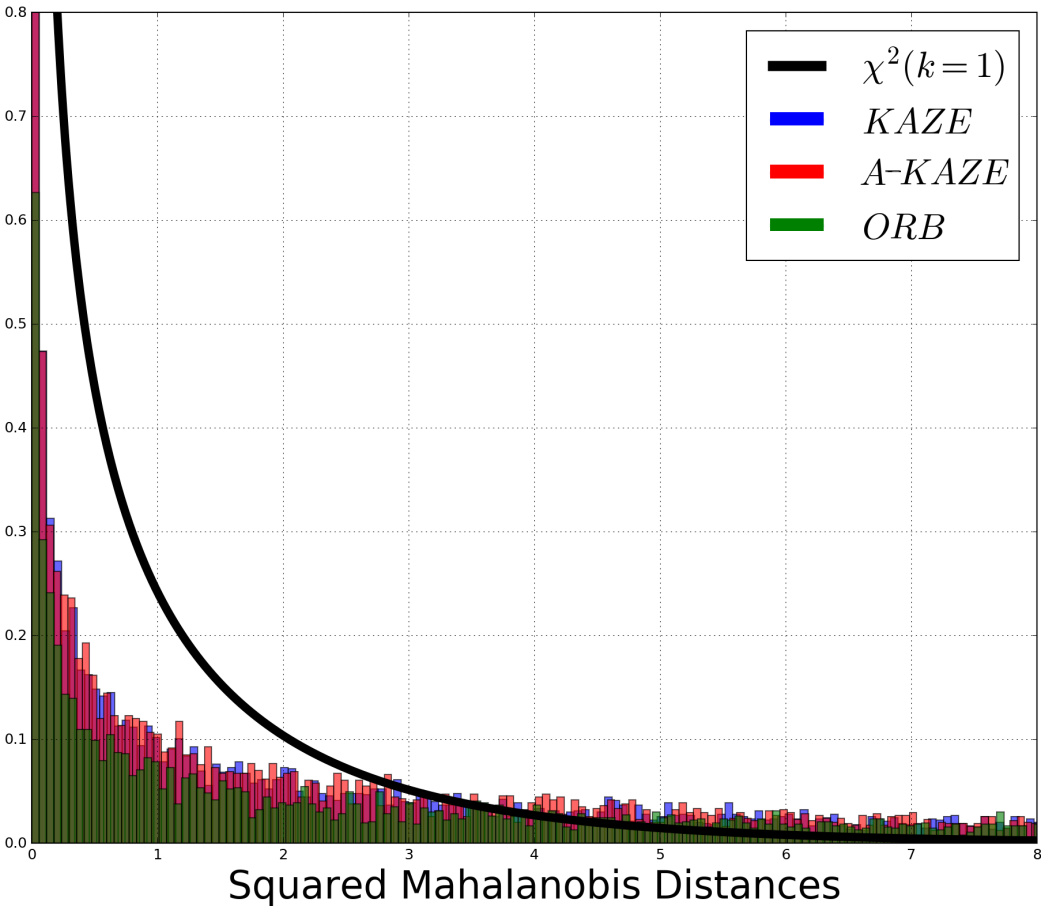
>> Performance

Positioning	L1	L1	L1, L2	L1, L2
Position accuracy (CEP)				
SPS	1.8 m	1.8 m	1.5 m	1.5 m
SBAS	0.6 m	0.6 m	0.6 m	0.6 m
DGPS	0.4 m	0.4 m	0.4 m	0.4 m
RTK			0.01 m	0.01 m
Velocity accuracy (RMS)	0.1 km/h	0.1 km/h	0.05 km/h	0.05 km/h
Roll/pitch accuracy (1σ)	0.05°	0.05°	0.03°	0.03°
Heading accuracy (1σ)[2]	0.1°	0.1°	0.1°	0.1°
Track angle accuracy (1σ)[3]	0.1°	0.1°	0.07°	0.07°
Slip angle accuracy (1σ)[4]	0.2°	0.2°	0.15°	0.15°
Dual antenna	x	✓	x	✓

APPENDIX B

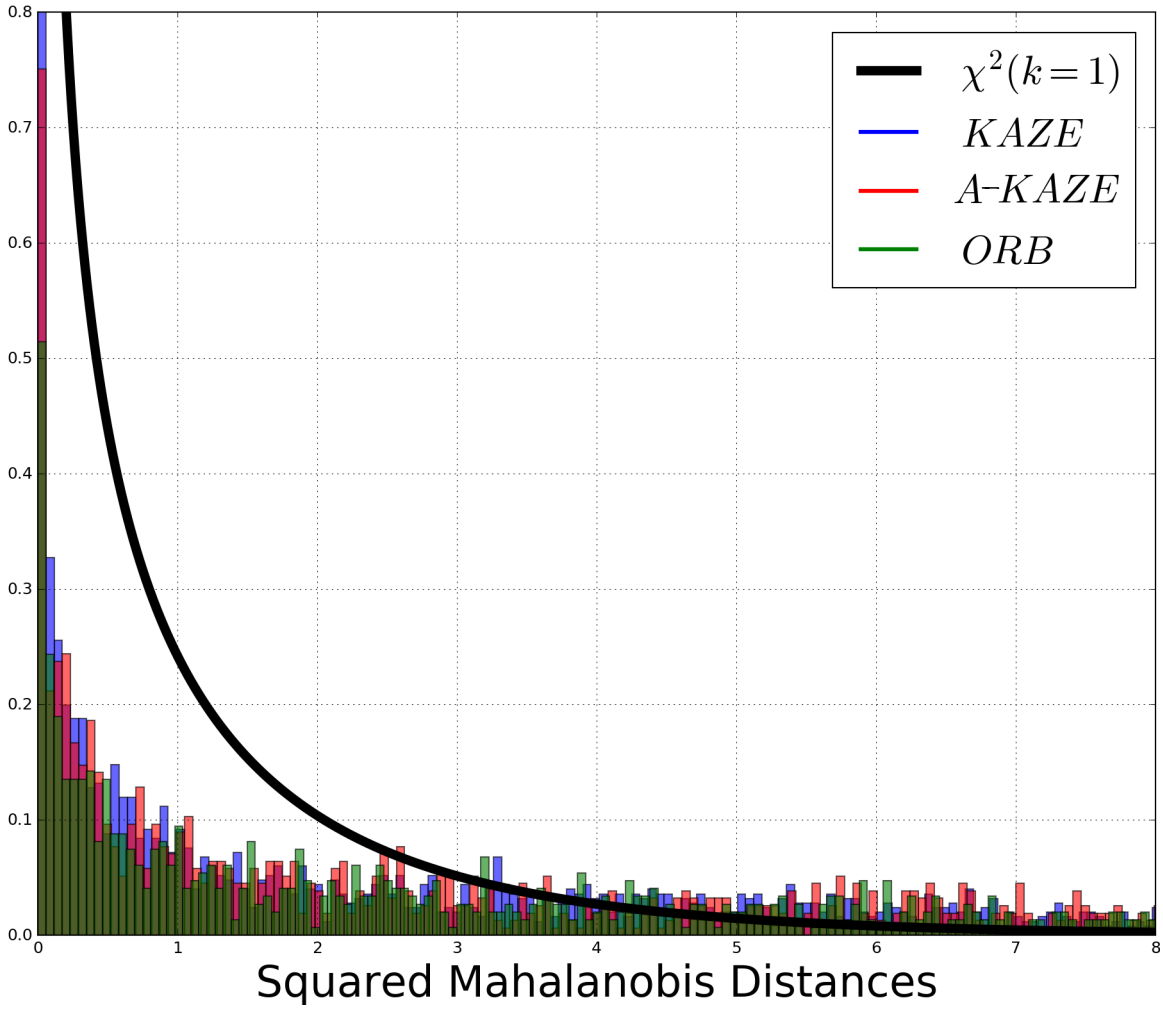
This Appendix contains the histogram Chi squared χ^2 distributions results. These distributions are of squared Mahalanobis distances obtained using the LiDAR measurements. Some of the distributions are a bit more noise due to lack of sample numbers. The distribution continues along the horizontal axis, but is zoomed-in for display purposes.

Dataset 0:



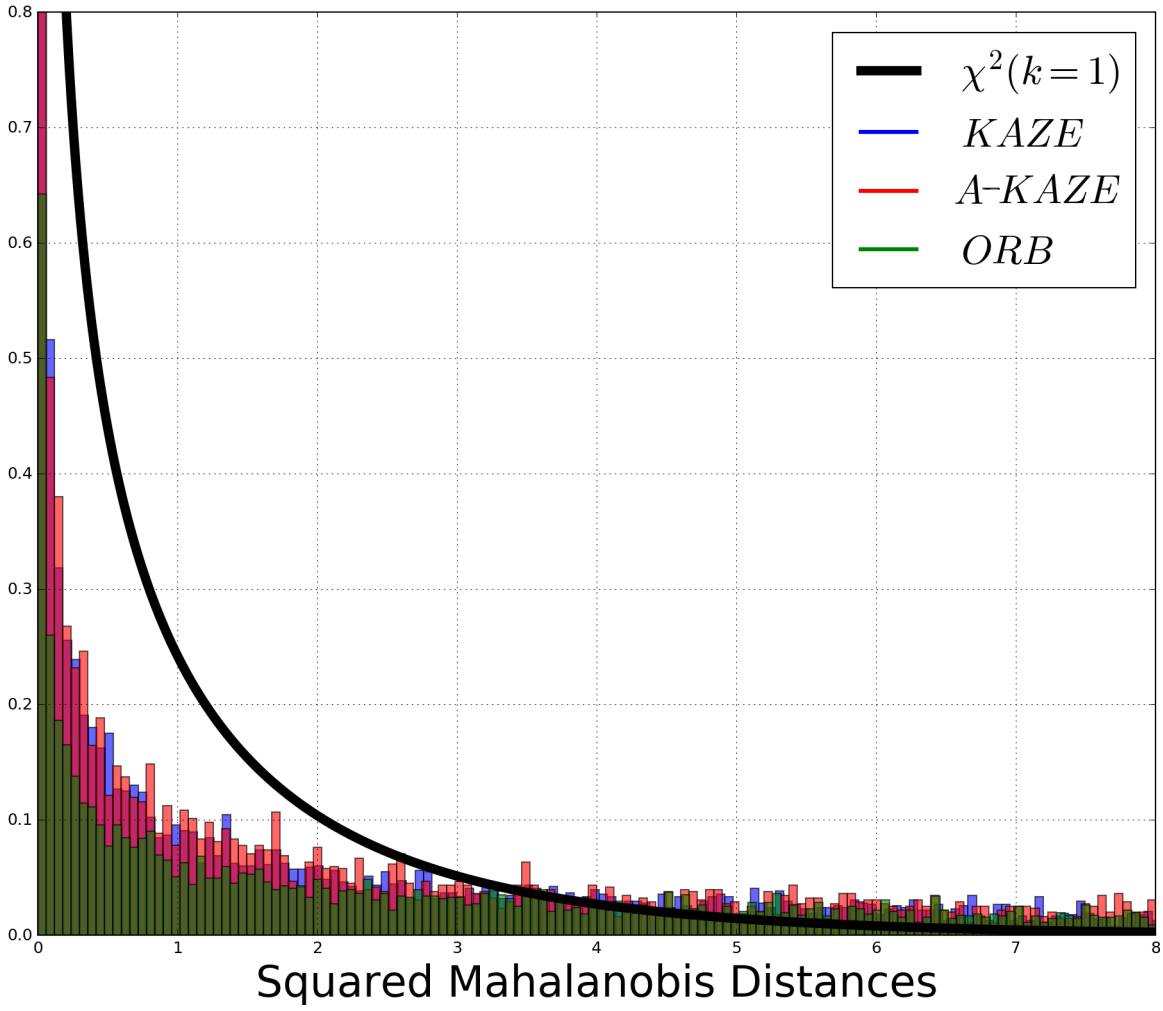
41324 samples

Dataset 1:



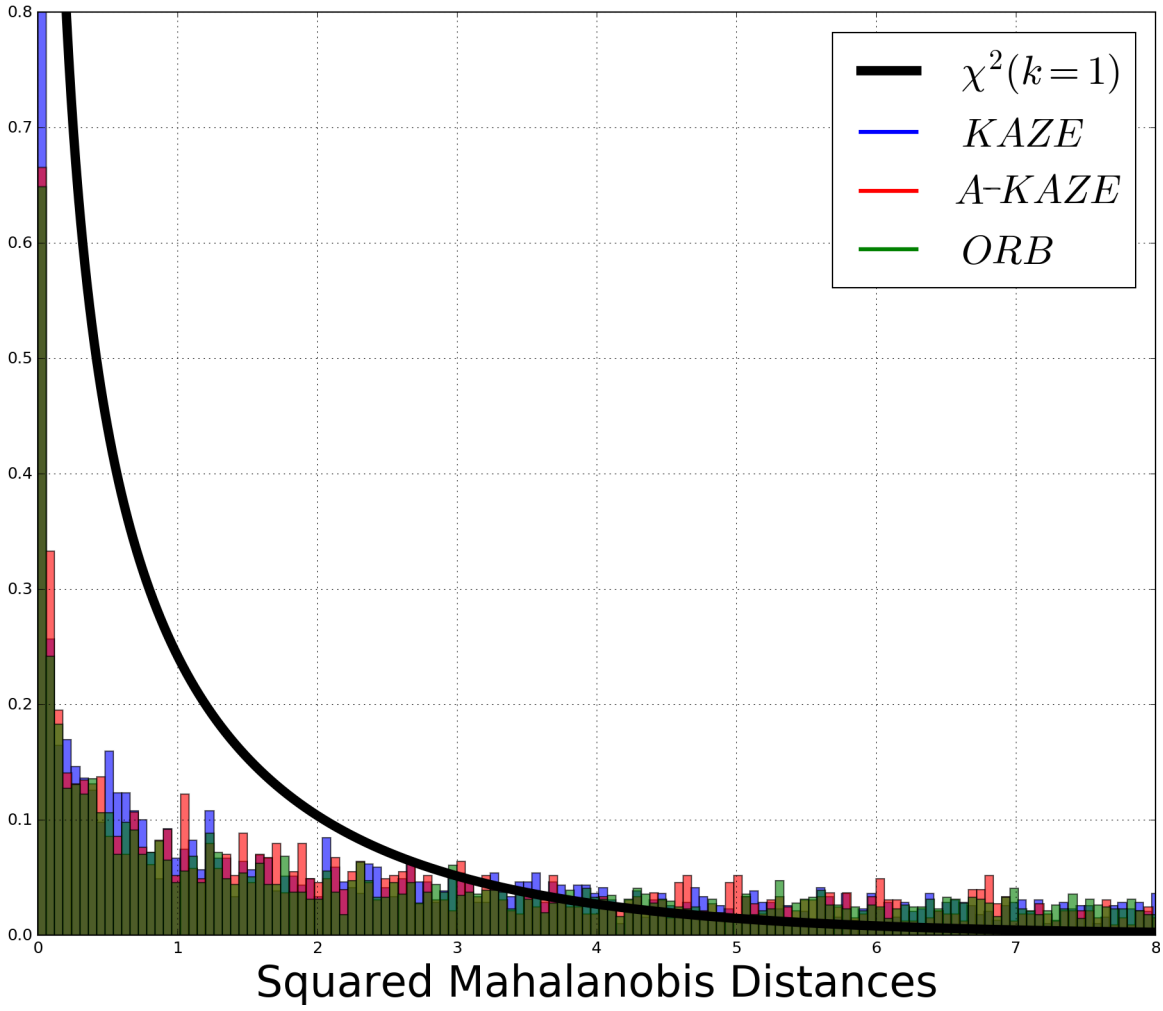
9651 samples

Dataset 2:



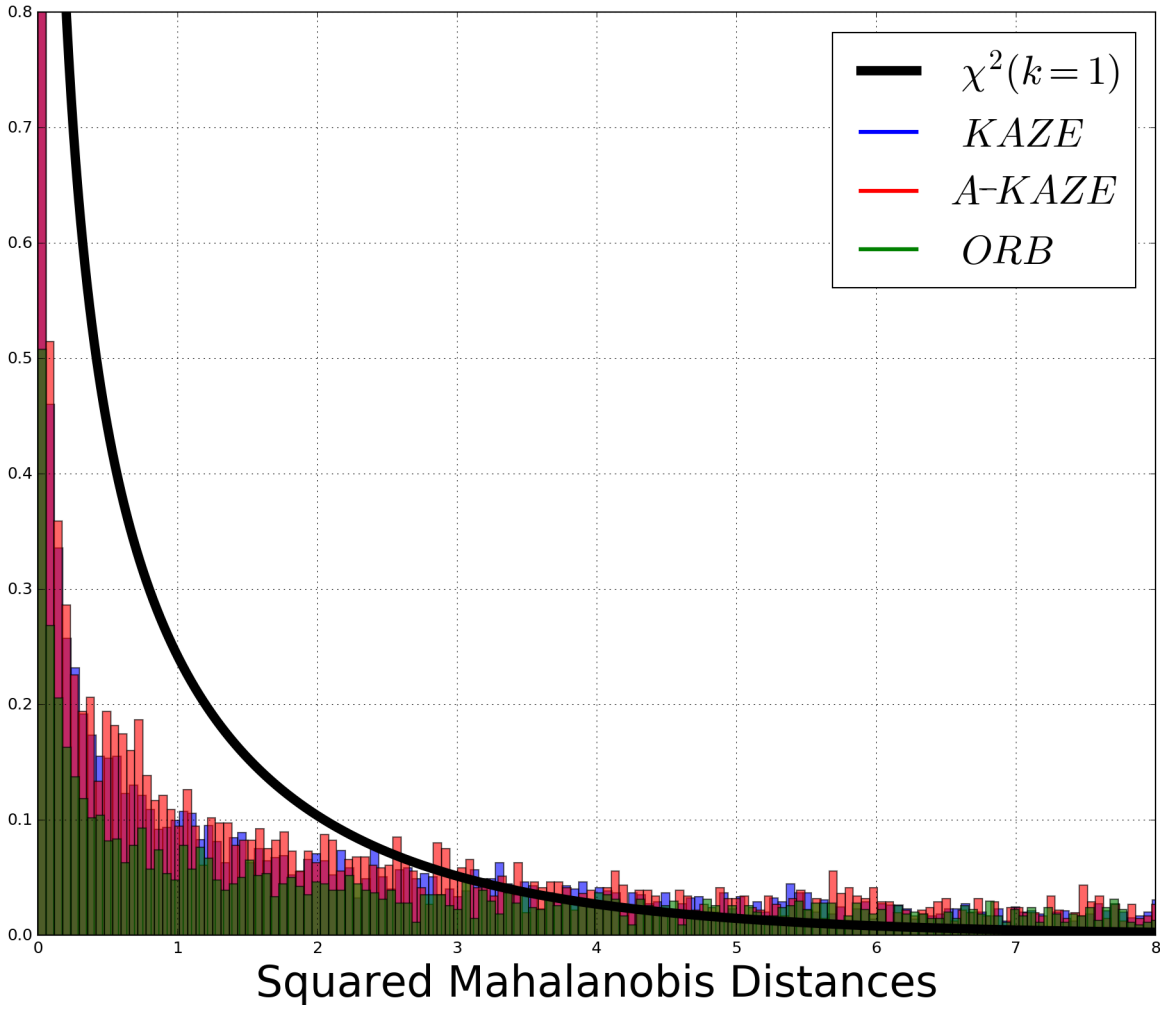
38126 samples

Dataset 3:



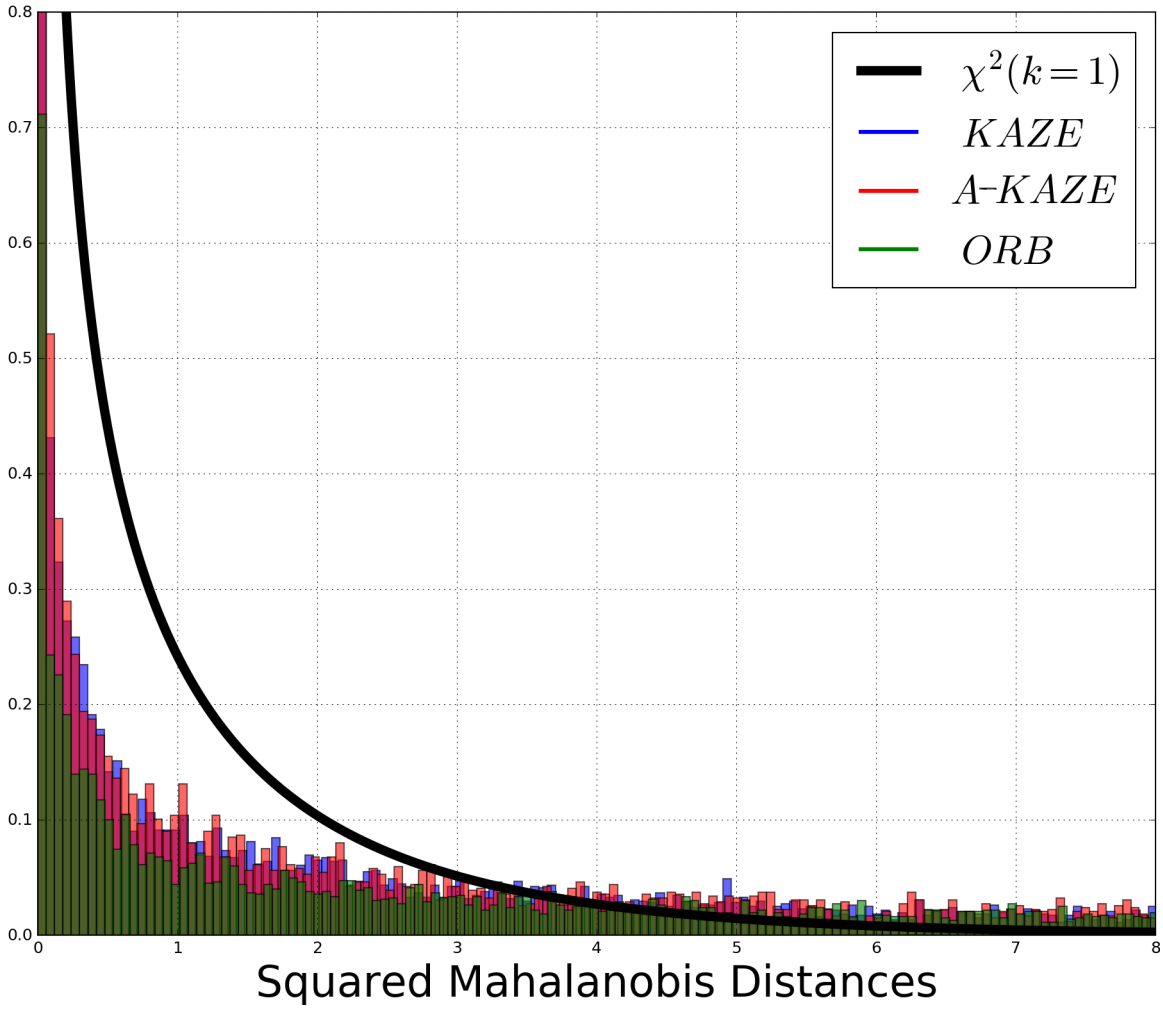
40959 samples

Dataset 4:



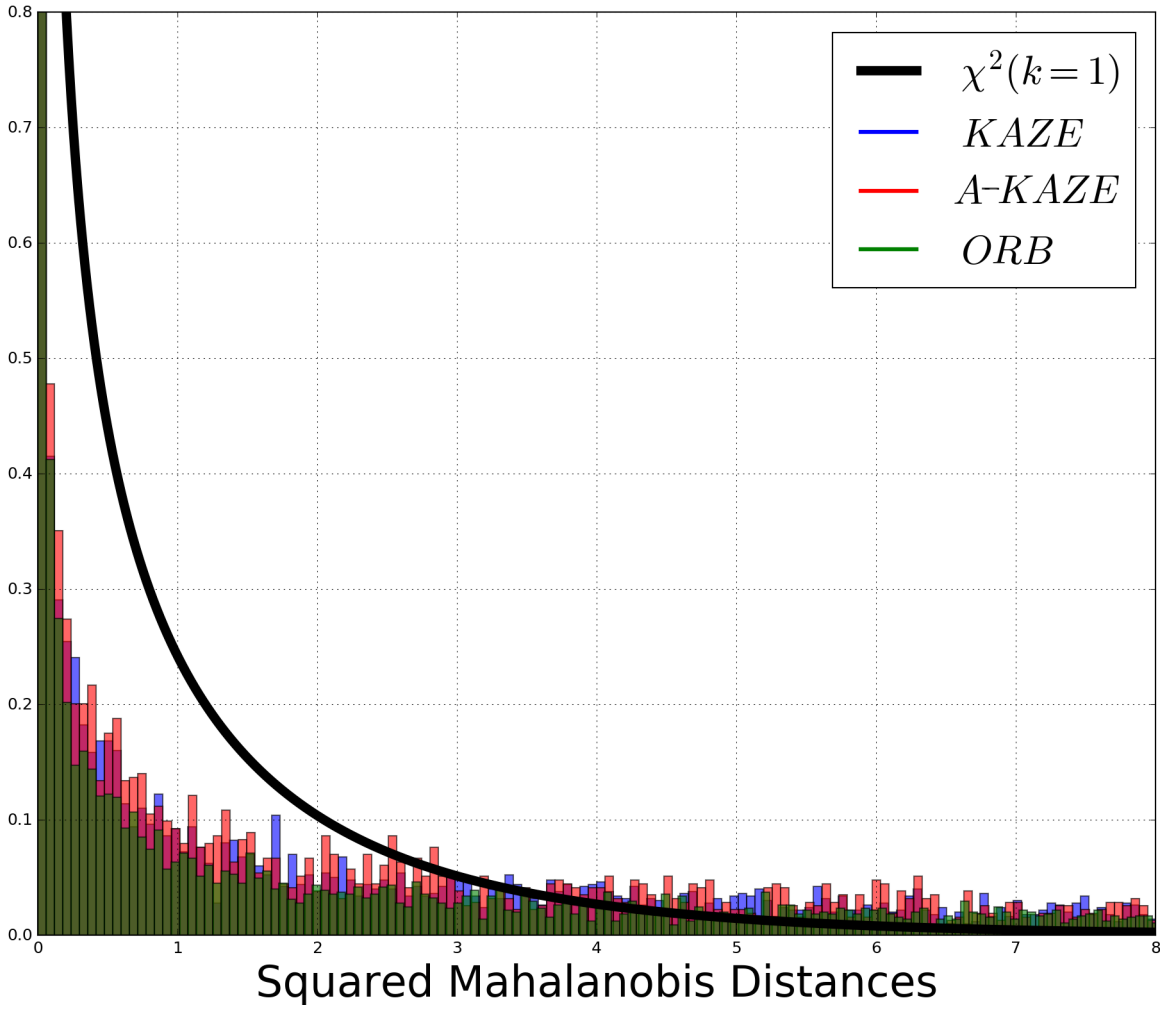
27677 samples

Dataset 5:



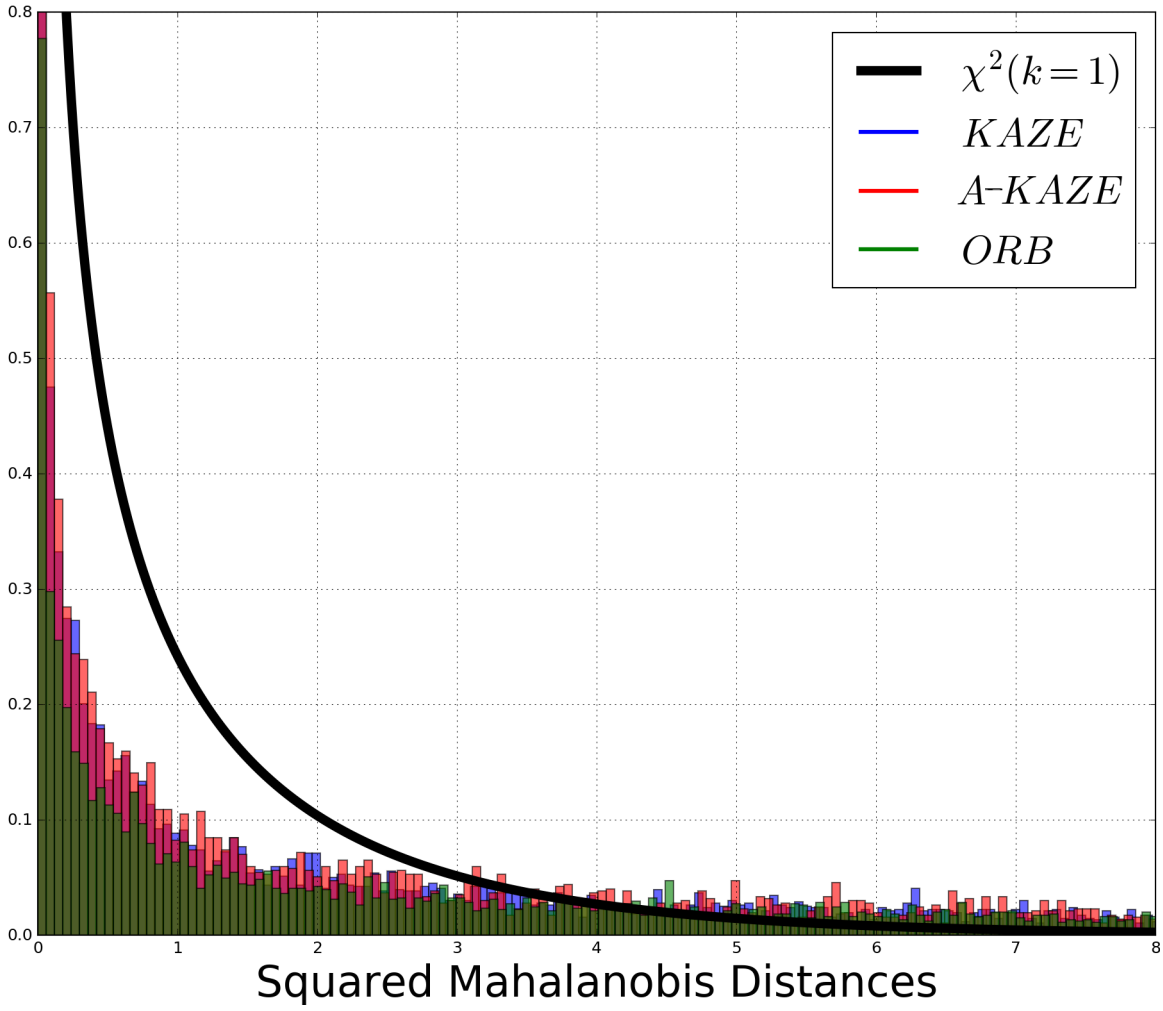
40251 samples

Dataset 6:



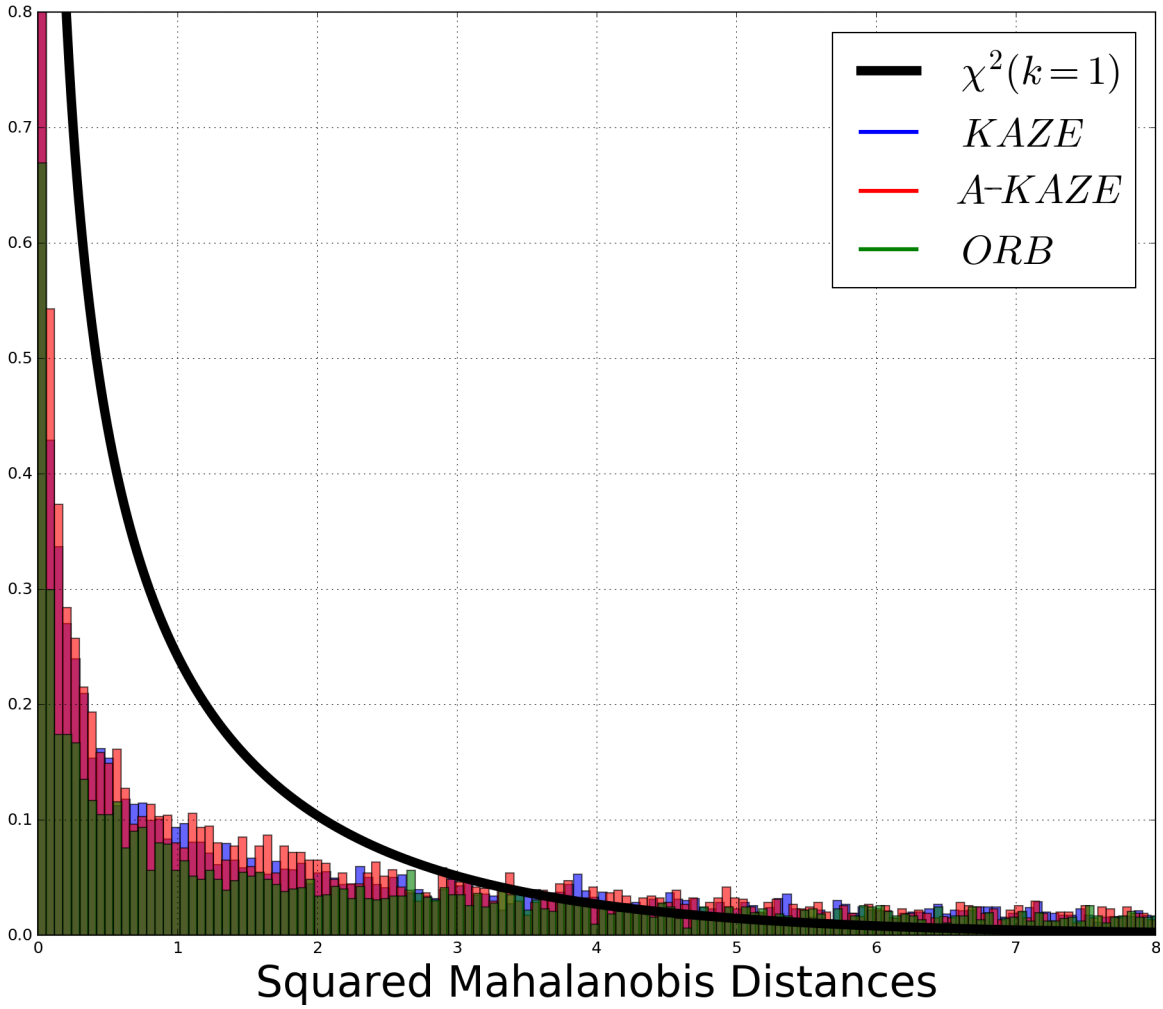
25066 samples

Dataset 7:



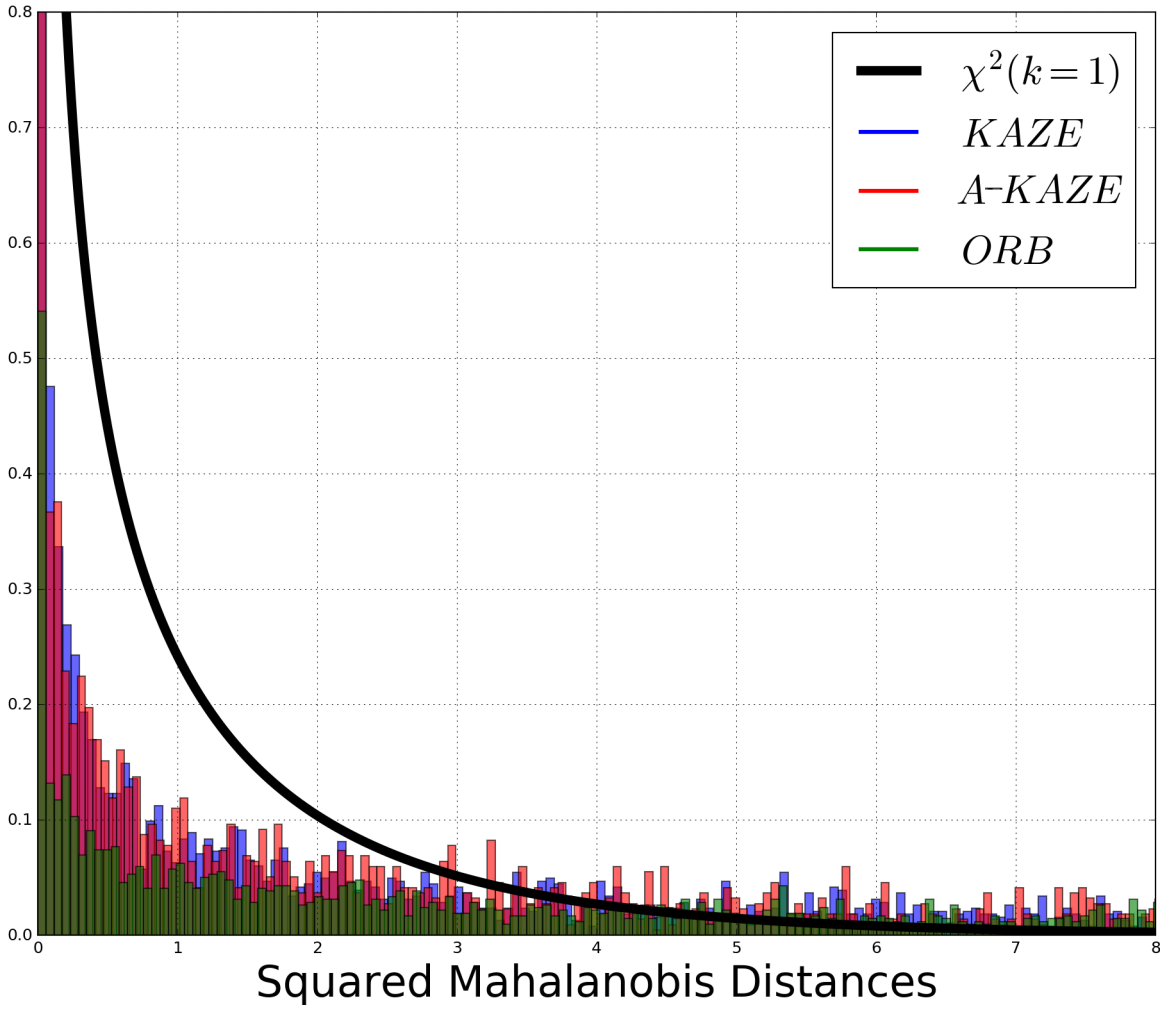
36778 samples

Dataset 8:



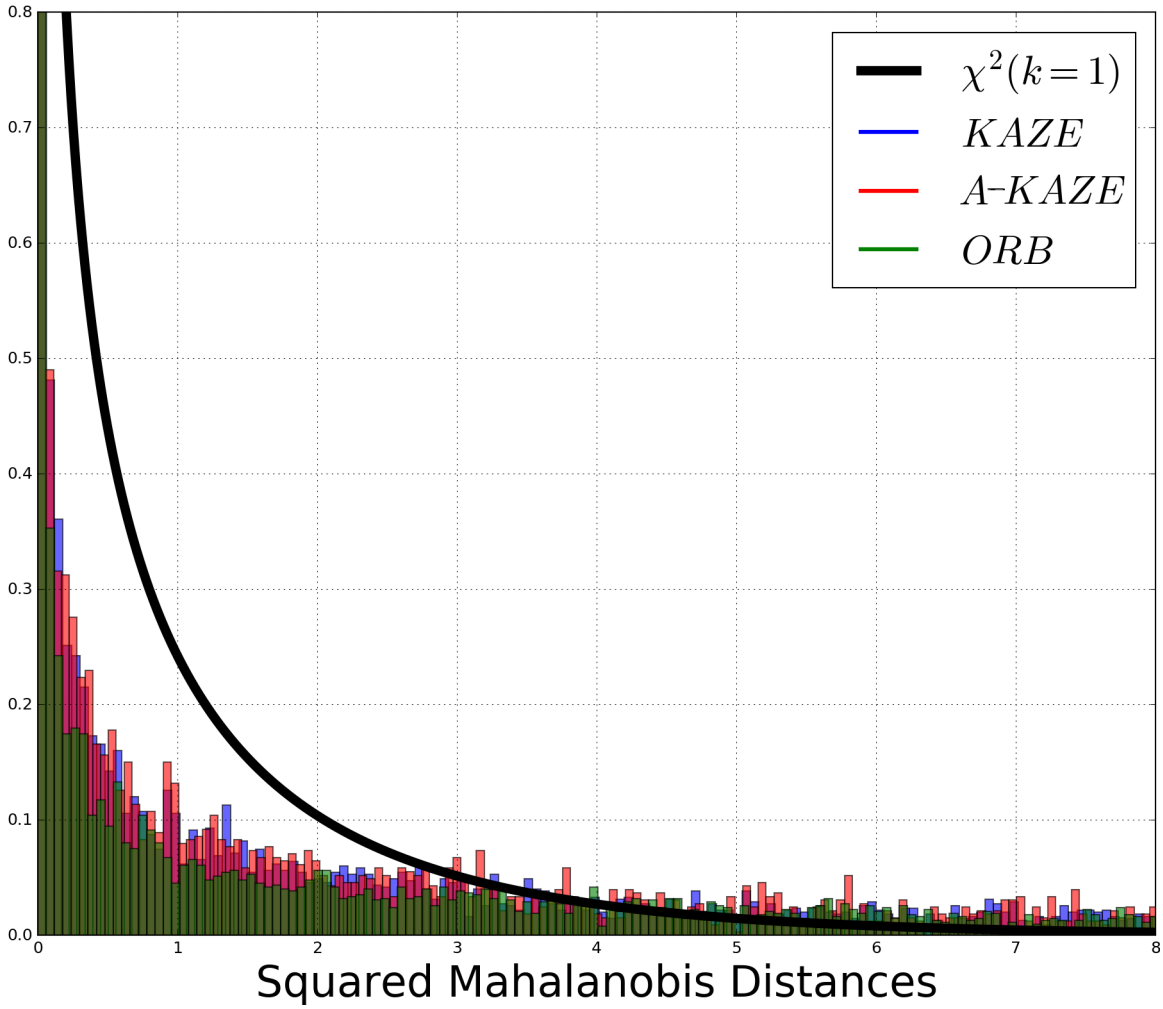
45821 samples

Dataset 9:



17568 samples

Dataset 10:



25990 samples

BIBLIOGRAPHY

[1] EECS 598-08, 'Basic Stereo and Epipolar Geometry', S Savarese. [Online]. Available: https://web.eecs.umich.edu/~jjcorso/t/598F14/files/lecture_1022_epipolar.pdf. [Accessed: 21- October- 2017].

[2] glumpy.readthedocs.io, 'Rendering a cube', Rougier. [Online]. Available: https://github.com/rougier/glumpy/blob/master/doc/_static/projection.svg. [Accessed: 28- October- 2017].

[3] HDL-64E, 'RT3000'. [Online]. Available: http://www.velodynelidar.com/lidar/products/manual/HDL-64E%20Manual.pdf. [Accessed: 21- October- 2017].

[4] http://blog.csdn.net, 'Dense Sift', 2014. [Online]. Available: http://blog.csdn.net/happyer88/article/details/46622657. [Accessed: 20- May- 2016].

[5] http://docs.opencv.org, 'FAST Algorithm for Corner Detection', 2014. [Online]. Available: http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_feature2d/py_fast/py_fast.html. [Accessed: 25- May- 2016].

[6] http://docs.opencv.org, 'Harris corner detector', 2016. [Online]. Available: http://docs.opencv.org/2.4/doc/tutorials/features2d/trackingmotion/harris_detector/harris_detector.html. [Accessed: 18- May- 2016].

[7] http://docs.opencv.org, 'Introduction to SIFT (Scale-Invariant Feature Transform)', 2014. [Online]. Available: http://docs.opencv.org/3.1.0/da/df5/tutorial_py_sift_intro.htmlgsc.tab=0. [Accessed: 20- May- 2016].

[8] http://docs.opencv.org, 'Introduction to SURF (Speeded-up Robust Features)', 2014. [Online]. Available: http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_feature2d/py_surf_intro/py_surf_intro.html. [Accessed: 23- May- 2016].

[9] http://ksimek.github.io, 'Dissecting the Camera Matrix, Part 2: The Extrinsic Matrix', Kyle Simek. [Online]. Available: http://ksimek.github.io/2012/08/22/extrinsic/. [Accessed: 26- July- 2017].

[10] OXTS, 'High Definition Lidar Sensor', Velodyne. [Online]. Available: http://www.techtarget.com.br/site/wp-content/uploads/2016/08/RT3000-brochure.pdf. [Accessed: 21- October- 2017].

[11] suinotes.files.wordpress.com, 'Scribbling...', 2010. [Online]. Available: https://suinotes.files.wordpress.com/2010/05/laplacian_of_gaussian.png. [Accessed: 20- May- 2016].

[12] www.juergenwiki.de, 'SURF (Speeded Up Robust Features)', 2014. [Online]. Available: http://www.juergenwiki.de/work/wiki/doku.php?id=public:surf. [Accessed: 23- May- 2016].

[13] A Bartoli A Davison, P Alcantarilla. Kaze features. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part VI*, ECCV'12, pages 214–227, Berlin, Heidelberg, 2012. Springer-Verlag.

[14] Pablo Alcantarilla, Jesus Nuevo, and Adrien Bartoli. Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces. *Procedings of the British Machine Vision Conference 2013*, pages 13.1–13.11, 2013.

[15] Pantelis Maroudis Panagiotis Lytrivis Angelos Amditis, George Thomaidis and Giannis Karaseitanidis. Multiple hypothesis tracking implementation [online]. available: http://www.intechopen.com/books/laser-scannertechnology/multiple-hypothesis-tracking-implementation, 2012.

[16] Nando de Freitas & Neil Gordon Arnaud Doucet, editor. *Sequential Monte Carlo Methods in Practice*. Springer, 2001.

[17] Y. Bar-Shalom. Extension of the probabilistic data association filter in multi-target tracking. *Proc. 5th Symp. on Nonlinear Estimation*, pages 61–21, Sept. 1974.

[18] Y. Bar-Shalom and T.E. Fortmann. *Tracking and Data Association*. Mathematics in Science and Engineering Series. Academic Press, 1988.

[19] Yaakov Bar-Shalom and Xiao Rong Li. Multitarget-Multisensor Tracking: Principles and Techniques. *IEEE Control Systems*, 16(1):93, 1996.

[20] Yaakov Bar-Shalom and Edison Tse. Tracking in a cluttered environment with probabilistic data association. *Automatica*, 11(5):451–460, September 1975.

[21] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer Vision Image Underst.*, 110(3):346–359, June 2008.

[22] R.J. Beyers and AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERING. *Joint Probability Data Association (JPDA) on Tracking Multiple Munitions Fragments*. Defense Technical Information Center, 1988.

[23] S.S. Blackman. *Multiple-target Tracking with Radar Applications*. Radar Library. Artech House, 1986.

[24] S.S. Blackman. Multiple hypothesis tracking for multiple target tracking. *IEEE Aerospace and Electronic Systems Magazine*, 19(1):5–18, 2004.

[25] Hsiang-Jen Chien, Chen-Chi Chuang, Chia-Yen Chen, and Reinhard Klette. When to use what feature? sift, surf, orb, or A-KAZE features for monocular visual odometry. In *IVCNZ*, pages 1–6. IEEE, 2016.

[26] Alexander Chiu, Thomas Jones, and Corne E. Van Daalen. A comparison of linearisation and the unscented transform for computer vision applications. *2016 Pattern Recognition Association of South Africa and Robotics and Mechatronics International Conference, PRASA-RobMech 2016*, (1):16–21, 2017.

[27] Kg Derpanis. The harris corner detector. *York University*, (March):2–3, 2004.

[28] Arnaud Doucet, Simon Godsill, and Christophe Andrieu. On sequential monte carlo sampling methods for bayesian filtering. *Statistics and Computing*, 10(3):197–208, July 2000.

[29] Olivier Faugeras, Quang-Tuan Luong, and T. Papadopoulou. *The Geometry of Multiple Images: The Laws That Govern The Formation of Images of A Scene and Some of Their Applications*. MIT Press, Cambridge, MA, USA, 2001.

[30] Michael Feldmann and Wolfgang Koch. Comments on "Bayesian approach to extended object and cluster tracking using random matrices". *IEEE Transactions on Aerospace and Electronic Systems*, 48(2):1687–1693, 2012.

[31] TE Fortmann, Y. Bar-Shalom, and M. Scheffe. Sonar tracking of multiple targets using joint probabilistic data association. *IEEE Journal of Oceanic Engineering*, 8(3):173–184, 1983.

[32] D. Franken, M. Schmidt, and M. Ulmke. "Spooky action at a distance" in the cardinalized probability hypothesis density filter. *IEEE Transactions on Aerospace and Electronic Systems*, 45(4):1657–1664, 2009.

[33] Botha Frik. Data Fusion of Radar and Stereo Vision for Detection and Tracking of Moving Objects. Master's thesis, University of Stellenbosch, South Africa, 2016.

[34] A Geiger, P Lenz, C Stiller, and R Urtasun. Vision meets robotics: The KITTI dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[35] Donald Bernard Gennery. *Modelling the Environment of an Exploring Vehicle by Means of Stereo Vision*. PhD thesis, Stanford, CA, USA, 1980. AAI8024660.

[36] Marius Goosen, Barend J. van Wyk, and Michael A. van Wyk. Survey of jpda algorithms for possible real-time implementation. 2004.

[37] Sven Grewenig, Joachim Weickert, and Andrés Bruhn. From box filtering to fast explicit diffusion. In *Pattern Recognition - 32nd DAGM Symposium, Darmstadt, Germany, September 22-24, 2010. Proceedings*, pages 533–542, 2010.

[38] Maridalia Guerrero. A Comparative Study of Three Image Matcing Algorithms : Sift , Surf , and Fast. *Image Rochester NY*, pages 1–93, 2011.

[39] R.W. Hamming. *Coding and Information Theory*. Pearson Education, Limited, 1980.

[40] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2 edition, 2003.

[41] Berthold K.P. Horn and Brian G. Schunck. Determining optical flow. Technical report, Cambridge, MA, USA, 1980.

[42] Robert J. Fitzgerald. Track biases and coalescence with probabilistic data association. AES-21:822 – 825, 12 1985.

[43] C. Loop and Zhengyou Zhang. Computing rectifying homographies for stereo vision. *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, pages 125–131, 1999.

[44] David G Lowe. Distinctive image features from scale invariant keypoints. *Int'l Journal of Computer Vision*, 60:91–11020042, 2004.

[45] Davide MacAgnano and Giuseppe Thadeu Freitas De Abreu. Adaptive gating for multi-target tracking with Gaussian mixture filters. *IEEE Transactions on Signal Processing*, 60(3):1533–1538, 2012.

[46] R. Mahler. *An Introduction to Multisource-multitarget Statistics and Applications*. Lockheed Martin, 2000.

[47] R. Mahler. A Theoretical Foundation for the Stein-Winter Probability Hypothesis Density (PhD) Multi-Target Tracking Approach. In *Proceedings of the 2000 MSS National Symposium on Sensor and Data Fusion*, 2002.

[48] R. Mahler. Multitarget Bayes filtering via first-order multitarget moments. *IEEE Transactions on Aerospace and Electronic Systems*, 39(4):1152–1178, 2003.

[49] Ronald Mahler. PHD filters of higher order in target number. *IEEE Transactions on Aerospace and Electronic Systems*, 43(4):1523–1543, 2007.

[50] Larry H. Matthies and Steven A. Shafer. Error modeling in stereo navigation. *IEEE J. Robotics and Automation*, 3(3):239–248, 1987.

[51] Rudolph Van Der Merwe. Sigma-point Kalman filters for probabilistic inference in dynamic state-space models. *PhD thesis*, (April):378, 2004.

[52] Ba ngu Vo and Wing kin Ma. The gaussian mixture probability hypothesis density filter. *IEEE Trans. SP*, pages 4091–4104, 2006.

[53] Frank Nielsen and Richard Nock. Clustering multivariate normal distributions. In *Emerging Trends in Visual Computing, LIX Fall Colloquium, ETVC 2008, Palaiseau, France, November 18-20, 2008. Revised Invited Papers*, pages 164–174, 2008.

[54] D. V. Papadimitriou and T. J. Dennis. Epipolar line estimation and rectification for stereo image pairs. *IEEE Transactions on Image Processing*, 5(4):672–676, 1996.

[55] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Trans. Pattern Anal. Mach. Intell.*, 12(7):629–639, July 1990.

[56] Anna Petrovskaya, Mathias Perrollaz, Luciano Oliveira, Luciano Spinello, Rudolph Triebel, Alexandros Makris, John-david Yoder, Christian Laugier, Urbano Nunes, and Pierre Bessiere. Awareness of Road Scene Participants for Autonomous Driving. 2011.

[57] Alessandro Pieropan, Mårten Björkman, Niklas Bergström, and Danica Kragic. Feature descriptors for tracking by detection: a benchmark. *CoRR*, abs/1607.06178, 2016.

[58] Michael Potmesil and Indranil Chakravarty. Synthetic Image Generation with a Lens and Aperture Camera Model. *ACM Transactions on Graphics*, 1(2):85–108, 1982.

[59] Donald B. Reid. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, 24:843–854, 1979.

[60] Angela Rojas, Antonio Calvo, and José Muñoz. A dense disparity map of stereo images. *Pattern Recognition Letters*, 18(4):385–393, 1997.

[61] Paul L. Rosin. Measuring corner properties. *Computer Vision and Image Understanding*, 73(2):291–307, 1999.

[62] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 3951 LNCS:430–443, 2006.

[63] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary R. Bradski. ORB: an efficient alternative to SIFT or SURF. In *IEEE International Conference on Computer Vision, ICCV 2011, Barcelona, Spain, November 6-13, 2011*, pages 2564–2571, 2011.

[64] Jianbo Shi and Carlo Tomasi. Good features to track. In *1994 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94)*, pages 593 – 600, January 1994.

[65] R. Smith, M. Self, and P. Cheeseman. Autonomous robot vehicles. chapter : Estimating Uncertain Spatial Relationships in Robotics, pages 167–193. Springer-Verlag New York, Inc., New York, NY, USA, 1990.

[66] Randall C. Smith and Peter Cheeseman. On the representation and estimation of spatial uncertainly. *I. J. Robotics Res.*, 5(4):56–68, December 1986.

[67] Christoph Strecha, Vincent Lepetit, Tomasz Trzcinski, Michael Calonder, Mustafa Özuysal, and Pascal Fua. Brief: Computing a local binary descriptor very fast. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34:1281–1298, 2011.

[68] Sebastian Thrun. Exploring artificial intelligence in the new millennium. chapter : Robotic Mapping: A Survey, pages 1–35. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

[69] Sundar Vedula, Simon Baker, Peter Rander, Robert Collins, and Takeo Kanade. Three-dimensional scene flow. 27(3):475 – 480, March 2005.

[70] D G Viswanathan. Features from Accelerated Segment Test (FAST). Available: `http://www.skybrary.aero/index.php/SeeandAvoid`. 2011.

[71] Ba Tuong Vo and Ba Ngu Vo. The Para-Normal Bayes Multi-Target Filter and the Spooky Effect. *Fusion2012*, pages 173–180, 2012.

[72] Ba-tuong Vo, Ba-ngu Vo, and Antonio Cantoni. Analytic Implementations of the Cardinalized Probability Hypothesis Density Filter Analytic Implementations of the Cardinalized Probability Hypothesis Density Filter. *IEEE Transactions on Signal Processing*, 55:3553–3567, 2007.

[73] Ba Tuong Vo, Ba Ngu Vo, and Antonio Cantoni. The cardinalized probability hypothesis density filter for linear Gaussian multi-target models. *2006 IEEE Conference on Information Sciences and Systems, CISS 2006 - Proceedings*, pages 681–686, 2007.

[74] Chieh-Chih Wang, C. Thorpe, and A. Suppe. Ladar-based detection and tracking of moving objects from a ground vehicle at high speeds. In *IEEE IV2003 Intelligent Vehicles Symposium. Proceedings (Cat. No.03TH8683)*, pages 416–421, June 2003.

[75] Chieh-Chih Wang, Charles E. Thorpe, and Sebastian Thrun. Online simultaneous localization and mapping with detection and tracking of moving objects: theory and results from a ground vehicle in crowded urban areas. In *ICRA*, pages 842–849. IEEE, 2003.

[76] Chieh-Chih Wang, Charles E. Thorpe, Sebastian Thrun, Martial Hebert, and Hugh F. Durrant-Whyte. Simultaneous localization, mapping and moving object tracking. *I. J. Robotics Res.*, 26(9):889–916, 2007.

[77] Joachim Weickert. Efficient image segmentation using partial differential equations and morphology. *Technical reports*, 00-03a, January 2000.

[78] Joachim Weickert, Sven Grewenig, Christopher Schroers, and Andrés Bruhn. Cyclic schemes for pde-based image analysis. *International Journal of Computer Vision*, 118(3):275–299, 2016.

[79] Joachim Weickert, Bart M. Ter Haar Romeny, and Max A. Viergever. Efficient and reliable schemes for nonlinear diffusion filtering. *IEEE Transactions on Image Processing*, 7:398–410, 1998.

[80] Joachim Weickert, Bart M. ter Haar Romeny, and Max A. Viergever. Efficient and reliable schemes for nonlinear diffusion filtering. *IEEE Trans. Image Processing*, 7(3):398–410, 1998.

[81] Xin Yang and Kwang-Ting Cheng. LDB: an ultra-fast feature for scalable augmented reality on mobile devices. In *11th IEEE International Symposium on Mixed and Augmented Reality, ISMAR 2012, Atlanta, GA, USA, November 5-8, 2012*, pages 49–57, 2012.

[82] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(11):1330–1334, November 2000.